Time Dependent Adjoint Methods for Single and Multi-disciplinary Problems

Dimitri Mavriplis *
Department of Mechanical Engineering
University of Wyoming
Laramie WY USA

September 14-17, 2015

Contents

*email:mavripl@uwyo.edu

1 Introduction

The general goal of a simulation-based analysis is to provide an accurate prediction of certain simulation objectives or quantities of interest for a given set of simulation inputs or parameters. As simulation capabilities have progressed, the ability to compute not only the simulation outputs as a function of inputs but also the derivatives or sensitivities of these outputs with respect to the inputs, has become increasingly recognized as an important capability for enabling sensitivity analysis, optimization, error estimation and uncertainty quantification. The simplest approach for computing sensitivities is to perturb the input parameters, rerun the simulation, and obtain the derivative of the outputs with respect to the perturbed input using a finite-difference approach. For non-linear problems, the accuracy of this approach is generally sensitive to the perturbation step size. A more consistent approach consists of deriving and solving the equations of the corresponding linearized problem about the current state. In both cases, these methods result in the sensitivities of all simulation outputs with respect to the perturbed input parameter, and must be repeated for each additional input parameter considered. On the other hand, solution of the corresponding adjoint problem (dual problem or reverse linearization) enables the calculation of the sensitivities of a single output with respect to all inputs. Thus, for a problem with an arbitrary number of inputs and outputs, the Jacobian matrix of output derivatives with respect to input parameters consists of a large rectangular matrix, and a single application of the forward sensitivity problem corresponds to the calculation of a column in this matrix, while a single adjoint solution delivers a matrix row.

Adjoint methods have gained widespread use in gradient-based optimization methods, since in many engineering optimization problems one is concerned with optimizing a single or small number of quantities of interest (e.g. performance objectives) given a large number of inputs or design parameters (e.g. shape parameters). Although adjoint methods are often discussed within the context of optimization, it is important to realize that an adjoint formulation is simply a tool for computing gradients which, in the general case, can be complementary to other techniques such as forward linearization approaches, depending on the structure of the sensitivity analysis problem, as mentioned above.

In general, continuous or discrete adjoint and forward sensitivity formulations are possible. In the continuous formulation, the continuous governing equations are first derived (by analytic differentiation and integration by parts for the adjoint problem) and then linearized, while in the discrete approach, the discretized equations are linearized directly (and transposed for the adjoint problem). For consistent discretizations, the two approaches converge to the same result with increasing discretization resolution. However, in general there is no guarantee that the discretization inherited in the discrete forward or adjoint approach is consistent, and substantial work to ensure so-called dual consistency has been performed in the literature [30, 4, 5, 27].

The use of adjoint methods is now fairly well established in steady-state aerodynamic shape optimization [7]. Additionally, adjoint methods for coupled aeroelastic or aerostructural optimization have been pursued over the last decade by various contributors [28, 13, 12, 9]. In spite of these advances, sensitivity analysis methods in general, and adjoint methods in particular have remained confined to a relatively small subset of simulation problems and there is a need to expand and apply these formulations to more

complex time-dependent and increasingly multidisciplinary problems. In these notes, we present techniques for deriving, implementing and applying forward sensitivity and adjoint methods to unsteady aerodynamic and aeroelastic problems. Although the basic formulations can be applied to continuous or discrete sensitivity approaches, in this work we focus on discrete sensitivity implementations. Our approach consists of building up exact analogues of each simulation component for complex simulation programs, such that these can be linked together using the same data-structures and reusing the same solvers developed for the analysis problem for complex time-dependent and coupled multidisciplinary problems. This methodical approach to building sensitivity analysis formulations is necessary for increasingly complex problems, and is facilitated by the discrete approach, which can be partially (or fully) automated, and verified exactly for correctness.

One of the most notable characteristics of adjoint methods for time-dependent problems is that, while the analysis problem is solved by integrating the governing equations forward in time, the adjoint formulation results in a backwards or reverse time integration starting at the final time step and ending at the initial physical time step. This is simply the result of a recurrence relation that is obtained when transposing the discrete forward sensitivity equations. However, for non-linear problems, since the equations must be linearized at the current state, the entire solution time-history (state at each time step) must be stored for use by the adjoint solver as it proceeds backwards in time. In general this is done by writing out the solution time history to disk as the analysis solver proceeds forward in time, and reading the solution values back in during the reverse adjoint procedure. Although substantial work has been done on mitigating the cost of this procedure, for problems of interest described here the I/O overhead is not substantial and as such additional techniques will not be considered herein.

In order to present a formal derivation of time-dependent forward and adjoint methods, we begin by deriving these formulations for a generic problem, and show how the adjoint problem can be derived using either simple chain-rule differentiation, or using Lagrange multipliers. We then focus on the particular structure of time-dependent problems and illustrate how the reverse time-integration relation arises within this simplified context. Subsequently, we derive the forward and adjoint formulations for a general time-dependent flow problem with a moving and deforming mesh. Use of the adjoint formulation for driving time-dependent optimization problems is then described. In section 4, we expand the formulation to general multidisciplinary time-dependent problems and subsequently focus on fully coupled aerodynamic and structural dynamic disciplines and demonstrate the use of the formulation for a coupled aero-elastic time-dependent optimization problem.

2 Forward and Adjoint Sensitivity Formulations

2.1 General Case

We consider a simulation which takes a set of input parameters D, and produces a set of outputs or quantities of interest L based on a computed state $\mathbf{U}$, where $\mathbf{U}$ is a field vector, for example a flow field solution for a computational fluid dynamics (CFD) problem. Thus, the functional dependence can be written as

$$L = L(\mathbf{U}(D), D) \tag{1}$$

where **U** depends implicitly on the inputs D, since it is obtained as the solution of the residual equation

$$\mathbf{R}(\mathbf{U}(D), D) = 0 \tag{2}$$

In order to compute the derivate of L with respect to D, we invoke the chain rule as

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial D} \tag{3}$$

$\frac{\partial \mathbf{U}}{\partial D}$ represents the flow field sensitivities which can be obtained by differentiating the residual equation or constraint $\mathbf{R}(\mathbf{U}(D), D) = 0$ as

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right] \frac{\partial \mathbf{U}}{\partial D} = -\frac{\partial \mathbf{R}}{\partial D} \tag{4}$$

Substituting this into equation (3) we obtain:

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} - \frac{\partial L}{\partial \mathbf{U}} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1} \frac{\partial \mathbf{R}}{\partial D} \tag{5}$$

The forward linearization approach corresponds to evaluating the last two terms on the right hand side first, while the adjoint approach consists of evaluating the second and third terms on the right-hand side first. Therefore, the forward linearization can be written as:

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right] \frac{\partial \mathbf{U}}{\partial D} = -\frac{\partial \mathbf{R}}{\partial D} \tag{6}$$

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial D} \tag{7}$$

whereas the adjoint procedure corresponds to:

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \mathbf{\Lambda} = -\frac{\partial L}{\partial \mathbf{U}}^T \tag{8}$$

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \mathbf{\Lambda}^T \frac{\partial \mathbf{R}}{\partial D} \tag{9}$$

In both of these formulations, most of the computational expense occurs in the solution of the first equation, which requires the inversion of a matrix of the size of the exact flow Jacobian. In the first case, a new solution of the flow sensitivity problem must be repeated for each input parameter D, whereas in the second case, a new adjoint solution is required for each objective L. Thus the forward sensitivity approach is best suited for cases with a single input and multiple objectives, while the adjoint approach is best suited for cases with a single objective and multiple input parameters.

The adjoint formulation can also be derived using Lagrange multipliers. In this approach, the residual equation $\mathbf{R}(\mathbf{U}(D), D) = 0$ is treated as a constraint, and an augmented functional J is formed as

$$J(\mathbf{U}(D), D) = L(\mathbf{U}(D), D) + \mathbf{\Lambda}^T \mathbf{R}(\mathbf{U}(D), D). \tag{10}$$

where we have introduced the Lagrange multiplier $\mathbf{\Lambda}$ which can be chosen arbitrarily, since it multiplies the vanishing residual constraint. Then, taking the derivatives of J with respect D, we obtain:

$$\frac{dJ}{dD} = \frac{dL}{dD} + \frac{\partial L}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial D} + \mathbf{\Lambda}^T(\frac{\partial \mathbf{R}}{\partial D} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial D}) \tag{11}$$

This expression may be rearranged by factoring out the flow sensitivity $\frac{\partial \mathbf{U}}{\partial D}$ as:

$$\frac{dJ}{dD} = \frac{\partial L}{\partial D} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial D} + (\frac{\partial L}{\partial \mathbf{U}} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial \mathbf{U}})\frac{\partial \mathbf{U}}{\partial D} \tag{12}$$

Since the flow sensitivity is expensive to compute (following equation (6)) and the Lagrange multiplier is arbitrary, we choose $\mathbf{\Lambda}$ such that the last term vanishes, ie.

$$\frac{\partial L}{\partial \mathbf{U}} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial \mathbf{U}} = 0 \tag{13}$$

which corresponds to the adjoint equation (8) and we recover the same expression for the final sensitivity

$$\frac{dJ}{dD} = \frac{\partial L}{\partial D} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial D} \tag{14}$$

noting that $\frac{dJ}{dD} = \frac{dL}{dD}$ since $\mathbf{R} = 0$.

Alternatively, we can derive the same equations by considering the Karush-Kuhn-Tucker (KKT) conditions [26] for the constrained optimization problem:

$$J_{min} = \min_{D \in I} J(D) \tag{15}$$

$$subject\ to \quad \mathbf{R}(\mathbf{U}(\mathbf{D}), \mathbf{D}) = 0 \tag{16}$$

by forming the augmented functional using Lagrange multipliers as previously

$$J(D, \mathbf{U}, \mathbf{\Lambda}) = L(\mathbf{U}(D), D) + \mathbf{\Lambda}^T\mathbf{R}(\mathbf{U}(D), D). \tag{17}$$

and requiring all derivatives of J to vanish at the optimum as:

$$\frac{\partial J}{\partial D} = \frac{\partial L}{\partial D} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial D} = 0 \tag{18}$$

$$\frac{\partial J}{\partial \mathbf{U}} = \frac{\partial L}{\partial \mathbf{U}} + \mathbf{\Lambda}^T\frac{\partial \mathbf{R}}{\partial \mathbf{U}} = 0 \tag{19}$$

$$\frac{\partial J}{\partial \mathbf{\Lambda}} = \mathbf{R}(\mathbf{U}) = 0 \tag{20}$$

The first equation gives the expression for the derivative $\frac{\partial J}{\partial D}$ and thus $\frac{dL}{dD}$ which must vanish at the optimum. The second equation corresponds to the adjoint equation and the third equation corresponds to the constraint equation, all of which must be satisifed at the constrained optimum.

2.2 Time-Dependent Formulation

In general, the above formulations are applicable to both steady-state and time-dependent problems. In the former case, the vector variables $(\mathbf{U}, \mathbf{\Lambda})$ and residual constraint equation apply over the spatial domain, and in the latter case, the vector variables span both space and time, and the residuals are integrated over all space and time. However, since time-dependent problems are usually solved by marching in time (either explicitly or implicitly), these equations have a particular structure in the time direction, and additional insight can be gained by writing out the analysis and resulting sensitivity formulations as time-stepping procedures. This can be performed using any of the approaches described above, although at this stage we will give an overview using the chain-rule approach to derive the forward sensitivity equations, and the KKT-Lagrange multiplier approach to derive the adjoint procedure, since these provide the simplest and most compact derivations, respectively.

We now consider a time-dependent simulation which produces a (time-integrated) objective L as a function of simulation inputs D. We also use lower case $\mathbf{u}$ to denote the spatial flow field at a particular time step. At each time step, the analysis procedure computes a new set of state variables $\mathbf{u}^n$ as a function of the state variables at the previous time step $n-1$ by solving the residual equations

$$\mathbf{R}^n(\mathbf{u}^n(D), \mathbf{u}^{n-1}(D), D) = 0 \tag{21}$$

Although this simple formulation assumes dependence only on the most recent $n-1$ time step values, more complex formulations will be considered in the following sections. The functional dependence for a general time-integrated objective may be written as:

$$L = L(\mathbf{u}^n(D), \mathbf{u}^{n-1}(D), \mathbf{u}^{n-2}(D), ..., \mathbf{u}^0(D), D) \tag{22}$$

Thus, using simple chain-rule differentiation, we obtain the sensitivity equation as:

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial \mathbf{u}^n}\frac{\partial \mathbf{u}^n}{\partial D} + \frac{\partial L}{\partial \mathbf{u}^{n-1}}\frac{\partial \mathbf{u}^{n-1}}{\partial D} + \frac{\partial L}{\partial \mathbf{u}^{n-2}}\frac{\partial \mathbf{u}^{n-2}}{\partial D} + ... + \frac{\partial L}{\partial \mathbf{u}^0}\frac{\partial \mathbf{u}^0}{\partial D} \tag{23}$$

As previously, the flow sensitivities (at each time step) can be computed by differentiating the residual constraint. For example, for time step n, the flow sensitivities can be obtained by solving

$$\left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n}\right]\frac{\partial \mathbf{u}^n}{\partial D} = -\left(\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}}\frac{\partial \mathbf{u}^{n-1}}{\partial D} + \frac{\partial \mathbf{R}^n}{\partial D}\right) \tag{24}$$

In the above equation, the flow sensitivities $\frac{\partial \mathbf{u}^n}{\partial D}$ at time step n can be computed provided the flow sensitivities at the previous time step $\frac{\partial \mathbf{u}^{n-1}}{\partial D}$ which appear on the right hand side are known. Thus the forward sensitivity problem can be solved in an analogous fashion to the analysis problem by marching forward in time, and solving a linearized flow problem at each time step (making use of sensitivities from previous time levels) to obtain the flow sensitivities at that time step. These can then be substituted into equation (23) to build up the final sensitivity of the overall time integrated objective. We note that if the initial

conditions $\mathbf{u}^0$ are prescribed, these do not depend on the parameters D, and the initial sensitivity equation is given as:

$$\left[\frac{\partial \mathbf{R}^1}{\partial \mathbf{u}^1}\right] \frac{\partial \mathbf{u}^1}{\partial D} = -\frac{\partial \mathbf{R}^1}{\partial D} \tag{25}$$

which corresponds to the sensitivity equation used to initiate the time-stepping procedure. On the other hand, if the time-dependent problem is initiated with a steady-state solution, then the initial sensitivity equation corresponds to an equation of the same form but evaluated at time level 0, and where the residual $\mathbf{R}^0(\mathbf{u}^0)$ corresponds to the residual of the steady-state problem used to compute the initial condition.

In order to derive the corresponding adjoint formulation for time-dependent problems, equations (24) may be used to substitute expressions for the flow sensitivities at each time step into equation (23), and terms may be regrouped and transposed as described previously for equations (5). Although we will derive the adjoint equations in subsequent sections in this manner, at this stage it is instructive to derive the time-dependent adjoint equations using the KKT-Lagrange multiplier approach since this provides a simpler and more compact derivation. Following this approach, we wish to minimize the objective L with functional form given by equation (22) subject to the constraints given by equation (21) at each time step. Since we have n constraints, we introduce n separate Lagrange multipliers $\mathbf{\Lambda}^n$ and construct the augmented functional J as:

$$\begin{aligned} J(D, \mathbf{u}^n, \mathbf{u}^{n-1}, ..., \mathbf{\Lambda}^n, \mathbf{\Lambda}^{n-1}, ...) &= L(\mathbf{u}^n(D), \mathbf{u}^{n-1}(D), \mathbf{u}^{n-2}(D), ..., \mathbf{u}^0(D), D) \\ &+ \mathbf{\Lambda}^{n^T} \mathbf{R}^n(\mathbf{u}^n(D), \mathbf{u}^{n-1}(D), D) \\ &+ \mathbf{\Lambda}^{n-1^T} \mathbf{R}^{n-1}(\mathbf{u}^{n-1}(D), \mathbf{u}^{n-2}(D), D) \\ &+ ... \\ &+ \mathbf{\Lambda}^{1^T} \mathbf{R}^1(\mathbf{u}^1(D), \mathbf{u}^0(D), D) \end{aligned}$$

As previously, taking the derivatives with respect to the Lagrange multipliers and setting these to zero we obtain the required constraints as:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{\Lambda}^n} &= \mathbf{R}^n(\mathbf{u}^n, \mathbf{u}^{n-1}, D) = 0 \\ \frac{\partial J}{\partial \mathbf{\Lambda}^{n-1}} &= \mathbf{R}^{n-1}(\mathbf{u}^{n-1}, \mathbf{u}^{n-2}, D) = 0 \\ &... \\ \frac{\partial J}{\partial \mathbf{\Lambda}^1} &= \mathbf{R}^1(\mathbf{u}^1, \mathbf{u}^0, D) = 0 \end{aligned} \tag{26}$$

assuming a prescribed initial condition $\mathbf{u}^0$. Taking the derivatives with respect to the

design parameters we obtain the sensitivity equation as:

$$\begin{aligned}\frac{\partial J}{\partial D} = \frac{\partial L}{\partial D} &+ \boldsymbol{\Lambda}^{n^T}\frac{\partial \mathbf{R}^n}{\partial D} \\ &+ \boldsymbol{\Lambda}^{n-1^T}\frac{\partial \mathbf{R}^{n-1}}{\partial D} \\ &+ \boldsymbol{\Lambda}^{n-2^T}\frac{\partial \mathbf{R}^{n-2}}{\partial D} \\ &+ \ldots = 0\end{aligned} \tag{27}$$

and taking the derivatives with respect to the state variables at each time step we obtain the adjoint equations as:

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{u}^n} &= \frac{\partial L}{\partial \mathbf{u}^n} + \boldsymbol{\Lambda}^{n^T}\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n} = 0 \\ \frac{\partial J}{\partial \mathbf{u}^{n-1}} &= \frac{\partial L}{\partial \mathbf{u}^{n-1}} + \boldsymbol{\Lambda}^{n^T}\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}} + \boldsymbol{\Lambda}^{n-1^T}\frac{\partial R^{n-1}}{\partial \mathbf{u}^{n-1}} = 0 \\ \frac{\partial J}{\partial \mathbf{u}^{n-2}} &= \frac{\partial L}{\partial \mathbf{u}^{n-2}} + \boldsymbol{\Lambda}^{n-1^T}\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-2}} + \boldsymbol{\Lambda}^{n-2^T}\frac{\partial \mathbf{R}^{n-2}}{\partial \mathbf{u}^{n-2}} = 0 \\ &\ldots \\ \frac{\partial J}{\partial \mathbf{u}^1} &= \frac{\partial L}{\partial \mathbf{u}^1} + \boldsymbol{\Lambda}^{2^T}\frac{\partial \mathbf{R}^2}{\partial \mathbf{u}^1} + \boldsymbol{\Lambda}^{1^T}\frac{\partial \mathbf{R}^1}{\partial \mathbf{u}^1} = 0\end{aligned} \tag{28}$$

These equations can be solved for the adjoint variables or Lagrange multipliers which can then be substituted into the sensitivity equation (e.g. equation (27)) in order to accumulate the final objective sensitivities. For example, the adjoint variable at the final time step n is obtained by solving an equation which has a similar form to equation (8) for the steady-state case as:

$$\left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n}\right]^T \boldsymbol{\Lambda}^n = -\frac{\partial L}{\partial \mathbf{u}^n}^T \tag{29}$$

whereas the adjoint variable at earlier time steps are obtained as (for the n-1 step shown here):

$$\left[\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-1}}\right]^T \boldsymbol{\Lambda}^{n-1} = -\frac{\partial L}{\partial \mathbf{u}^{n-1}}^T - \boldsymbol{\Lambda}^{n^T}\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}} \tag{30}$$

Thus the adjoint solution at the time level n-1 depends on the adjoint solution at time level n, and therefore equation (29) must be solved prior to solving equation (30). These equations correspond to a recurrence relation in time, which must be solved starting at the final time level n, and proceeding backwards in time until the initial time level is solved. This corresponds to the reverse of the procedure derived for the forward sensitivity described previously in equations (24) above, and occurs as a result of transposing the equations in the adjoint case.

3 Discrete Adjoint Formulation for General Unsteady Flow Problems

3.1 Analysis Formulation

The derivations in the previous section are purposely simplified for cases with simple functional dependencies. In this section we derive the general form for the forward and adjoint sensitivity equations for general unsteady flow problems with moving and deforming meshes. Mesh deformation can occur as a result of geometry shape changes, for example in cases where the input parameters correspond to shape design parameters, as well as due to prescribed time-dependent motion of the geometry both of which may require interior mesh deformation in order to avoid invalid mesh cells. Additionally, commonly used time-stepping schemes generally result in more complex time-dependent functional residual forms than those employed in the previous section. Although any of the previously described methods can be used to derive the forward and adjoint sensitivity equations, we will restrict ourselves to the chain-rule derivation approach, since this derivation requires only basic knowledge of the fundamental chain rule of differential calculus. Perhaps more importantly, the chain-rule derivation follows most closely the actual implementation of a discrete forward or adjoint sensitivity implementation and therefore provides an opportunity to include implementation details as part of the derivation.

3.1.1 Governing Equations of the Flow Problem in ALE Form

In order to perform the derivation, we must first consider the functional forms of the objectives and residuals for time-dependent moving mesh problems. In general, we are interested in solving the (Reynolds-averaged) Navier-Stokes equations in arbitrary Lagrange-Eulerian (ALE) form using finite-volume type discretizations. The conservative form of the Navier-Stokes equations is used in solving the flow problem. These may be written as:

$$\frac{\partial \mathbf{u}(\mathbf{x},t)}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{u}) = 0 \tag{31}$$

Applying the divergence theorem and integrating over a moving control volume $V(t)$ with boundary B yields:

$$\int_{V(t)} \frac{\partial \mathbf{u}}{\partial t} dV + \int_{dB(t)} \mathbf{F}(\mathbf{u}) \cdot \mathbf{n} dB = 0 \tag{32}$$

where $\mathbf{n}$ is the outwards surface normal vector. Using the differential identity:

$$\frac{\partial}{\partial t} \int_{V(t)} \mathbf{u} dV = \int_{V(t)} \frac{\partial \mathbf{u}}{\partial t} dV + \int_{dB(t)} \mathbf{u}(\dot{\mathbf{x}} \cdot \mathbf{n}) dB \tag{33}$$

equation (32) is rewritten as:

$$\frac{\partial}{\partial t} \int_{V(t)} \mathbf{u} dV + \int_{dB(t)} [\mathbf{F}(\mathbf{u}) - \dot{\mathbf{x}}\mathbf{u}] \cdot \mathbf{n} dB = 0 \tag{34}$$

or when considering cell-averaged values for the state $\mathbf{u}$ as:

$$\frac{\partial V\mathbf{u}}{\partial t} + \int_{dB(t)} [\mathbf{F}(\mathbf{u}) - \dot{\mathbf{x}}\mathbf{u}] \cdot \mathbf{n} dB = 0 \tag{35}$$

This is the Arbitrary-Lagrangian-Eulerian (ALE) finite-volume form of the governing equations. The equations are required in ALE form since the problem involves deforming meshes where mesh elements may change in shape and size at each time-step. Here V refers to the area of the control volume, $\dot{\mathbf{x}}$ is the vector of mesh face or edge velocities, and $\mathbf{n}$ is the unit normal of the face. $\mathbf{u}$ denotes the state vector of conserved variables and the flux vector $\mathbf{F}$ contains both inviscid and viscous fluxes.

3.1.2 Discretization and Solution Strategies

The spatial discretization relies on a vertex-based median-dual control volume formulation which is second-order accurate in space [20]. The time derivative term in the governing equations is discretized using either the first-order accurate backward-difference formula (BDF1):

$$\frac{\partial \mathbf{V}\mathbf{u}}{\partial t} = \frac{\mathbf{V}^n\mathbf{u}^n - \mathbf{V}^{n-1}\mathbf{u}^{n-1}}{\Delta t} \tag{36}$$

or a second-order accurate backwards difference formula (BDF2) requiring storage of an additional previous time level:

$$\frac{\partial \mathbf{V}\mathbf{u}}{\partial t} = \frac{\frac{3}{2}\mathbf{V}^n\mathbf{u}^n - 2\mathbf{V}^{n-1}\mathbf{u}^{n-1} + \frac{1}{2}\mathbf{V}^{n-2}\mathbf{u}^{n-2}}{\Delta t} \tag{37}$$

where the index n is used to indicate the current time-level, and $n-1$ and $n-2$ refer to previous time levels. $\mathbf{V}$ refers to the vector of mesh control volumes, and is indexed by the time level n, since for dynamic mesh problems these values change at each time step.

Denoting the spatially discretized terms at time level n by the operator $\mathbf{S}^n(\mathbf{u}^n)$, the resulting system of non-linear equations to be solved at each time step can be denoted as (shown for the case of the BDF1 scheme for simplicity):

$$\mathbf{R}^n = \frac{\mathbf{V}^n\mathbf{u}^n - \mathbf{V}^{n-1}\mathbf{u}^{n-1}}{\Delta t} + \mathbf{S}^n\left(\mathbf{u}^n\right) = 0 \tag{38}$$

3.1.3 The Discrete Geometric Conservation Law (GCL)

For moving mesh problems, new mesh coordinates values $\mathbf{x}^\mathbf{n}$ are obtained at each time step, either by prescribed motion (e.g. mesh rotation), by the solution of mesh deformation equations, or by a combination of both techniques. While these values are required to generate the new mesh control volumes and other metrics at the new time level, the governing equations in ALE form also require the determination of the mesh velocities $\dot{\mathbf{x}}$, as seen from equation (35). The determination of these terms must be considered carefully in order to maintain discrete conservation in ALE form. This property is embodied in the Geometric Conservation Law (GCL), which states that a uniform flow must remain an exact solution to the discretized equations in the presence of arbitrary mesh deformation [34, 3, 21]. Substituting $\mathbf{u} = constant$ into equation (35), and noting that the integral of the flux term $\mathbf{F}(\mathbf{u})$ in this case vanishes around a closed boundary, the mathematical statement of the GCL becomes:

$$\frac{\partial \mathbf{V}}{\partial t} - \int_{dB(t)} \dot{\mathbf{x}} \cdot \mathbf{n} dB = 0 \tag{39}$$

which provides a constraint on the method of evaluation of the grid velocity terms, and which depends on the specific time discretization scheme chosen for the left hand side of this equation. In reference [21] a method for constructing the required grid velocities which discretely satisfy equation (39) is developed for BDF schemes as well as implicit Runge-Kutta time discretizations. In the case of BDF schemes, the general difference form of the BDF time discretization is first written as a sequence of incremental differences between neighboring time levels and the face-integrated grid velocities are then individually equated to the volume swept by each control volume face between neighboring time levels. The exact calculation of the volume swept by a mesh face in three-dimensions requires the use of a two point quadrature rule between each neighboring time level [3, 21]. Thus, for BDF1 schemes this results in a two point time-integration rule, while for BDF2 schemes this results in a four point time-integration rule. However, for BDF1 schemes, the final functional dependence of the face-integrated grid velocities depends only on the coordinate values $\mathbf{x}^n$ and $\mathbf{x}^{n-1}$, while for BDF2 schemes, these values depend on $\mathbf{x}^n$, $\mathbf{x}^{n-1}$ and $\mathbf{x}^{n-2}$. Note that in this formulation, it is the face-integrated grid velocities (i.e. the right-hand side of equation (39)) and not the grid velocities themselves which are computed and used in the ALE formulation. Therefore, in the remainder of these notes, we use the notation $\dot{\mathbf{x}}$ to denote the face-integrated values in the place of the actual grid point velocities, for simplicity.

3.1.4 Functional Form of Unsteady Residual

The determination of the functional form of the unsteady residual is important for the linearization to be undertaken in the formulation of the adjoint problem discussed subsequently. By definition, for backwards difference schemes, the spatial discretization terms are to be evaluated exclusively at the most recent time level n. Noting that the volumes $\mathbf{V}$ are functions of the grid coordinates, we obtain for BDF2:

$$\mathbf{R}^n = \frac{3}{2\Delta t}\mathbf{V}(\mathbf{x}^n)\mathbf{u}^n - \frac{2}{\Delta t}\mathbf{V}(\mathbf{x}^{n-1})\mathbf{u}^{n-1} + \frac{1}{2\Delta t}\mathbf{V}(\mathbf{x}^{n-2})\mathbf{u}^{n-2} + \mathbf{S}^n(\mathbf{u}^n, \mathbf{x}^n, \dot{\mathbf{x}}^n) \tag{40}$$

However, given the functional form of face-integrated grid velocities discussed above, the spatial residual ends up with dependences on previous time levels as well, and the entire unsteady residual functional dependence becomes:

$$\mathbf{R}^n(\mathbf{u}^n, \mathbf{u}^{n-1}, \mathbf{u}^{n-2}, \mathbf{x}^n, \mathbf{x}^{n-1}, \mathbf{x}^{n-2}) = \tag{41}$$

$$\frac{3}{2\Delta t}\mathbf{V}(x^n)\mathbf{u}^n - \frac{2}{\Delta t}\mathbf{V}(x^{n-1})\mathbf{u}^{n-1} + \frac{1}{2\Delta t}\mathbf{V}(x^{n-2})\mathbf{u}^{n-2} + \mathbf{S}^n(\mathbf{u}^n, \mathbf{x}^n, \mathbf{x}^{n-1}, \mathbf{x}^{n-2}) \tag{42}$$

with a corresponding expression depending only on n and $n-1$ values for the BDF1 case.

3.1.5 Mesh Deformation Strategy

For time-dependent problems that are restricted to rigid body motion, the mesh motion at each time step can be prescribed through a series of solid body rotations and translations. However, for problems with deformable geometries and/or relative motion, the interior mesh must be deformed in response to the prescribed surface displacements in order to maintain a valid mesh at all time steps. In the context of a shape optimization problem,

interior mesh deformation may also be driven by surface displacements that arise in response to surface geometry design changes which occur throughout the design process for both steady-state and time-dependent problems. In general, the new mesh coordinates are obtained by solving a set of mesh deformation equations denoted as:

$$\mathbf{G}(\mathbf{x}, \mathbf{x_s}) = 0 \tag{43}$$

where $\mathbf{G}$ represents the residual of the mesh deformation equations, $\mathbf{x}$ represents the required mesh coordinates, and $\mathbf{x_s}$ represents the prescribed surface mesh coordinates. Although this generic notation covers both linear and non-linear formulations, in practice we make use of a linear elastic mesh motion approach. In this approach, the mesh is modeled as a linear elastic solid with a variable modulus of elasticity that can be prescribed either as inversely proportional to cell volume or to the distance of each cell from the nearest wall. In this case, the linear system that relates the interior vertex displacements in the mesh to known displacements on the boundaries is given as:

$$[K]\delta\mathbf{x} = \delta\mathbf{x_s} \tag{44}$$

where $\delta\mathbf{x}$ represents the mesh point displacements computed in response to the prescribed surface displacements $\delta\mathbf{x_s}$. For linear mesh deformation formulations, the stiffness matrix [K] is a sparse block matrix that is based on the initial configuration of the mesh and remains unchanged through the time-integration process.

3.2 Sensitivity Formulation

3.2.1 Steady-State Problems

Although the application of the above procedure to steady-state shape-optimization problems is well known, the resulting procedure is outlined in this section for completeness prior to discussing the time-dependent formulation. In this case, the simulation objective is only a function of the steady-state flow variables and grid point coordinates, and the sensitivity vector can be written using the chain rule as:

$$\frac{dL}{dD} = \frac{\partial L}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial D} + \frac{\partial L}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial D} \tag{45}$$

The last term corresponds to the flow variable sensitivities, which are determined by the constraint

$$\mathbf{R}\left(\mathbf{u}(D), \mathbf{x}(D)\right) = 0 \tag{46}$$

which merely states that the flow equation residuals must vanish at steady state. Linearization of equation (46) provides the expression for the flow sensitivities:

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right]\frac{\partial \mathbf{u}}{\partial D} = -\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial D} \tag{47}$$

However, solution of this equations requires that the mesh sensitivities $\frac{\partial \mathbf{x}}{\partial D}$ be available. Therefore, the mesh sensitivity equation, which is obtained by differentiating the mesh motion equation as:

$$[K]\frac{\partial \mathbf{x}}{\partial D} = \frac{\partial \mathbf{x_s}}{\partial D} \tag{48}$$

must first be solved. Substituting equations (47) and (48) into equation (45) we obtain the final form for the forward sensitivity equation:

$$\frac{dL}{dD} = \left[\frac{\partial L}{\partial \mathbf{x}} - \frac{\partial L}{\partial \mathbf{u}}\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right]^{-1}\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right][K]^{-1}\frac{\partial \mathbf{x}_s}{\partial D} \tag{49}$$

In practice, this equation is solved by first solving the mesh sensitivity equation (e.g. equation (48)), evaluating the right hand side of the flow sensitivity equation (e.g. equation (47)) and solving this equation, and finally substituting the computed mesh and flow sensitivities into equation (45) in order to obtain the final objective sensitivity.

For the adjoint problem, equation (49) is transposed to obtain

$$\frac{dL}{dD}^T = \frac{\partial \mathbf{x}_s}{\partial D}^T [K]^{-T}\left[\frac{\partial L}{\partial \mathbf{x}}^T - \frac{\partial \mathbf{R}}{\partial \mathbf{x}}^T\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right]^{-T}\frac{\partial L}{\partial \mathbf{u}}^T\right] \tag{50}$$

If we define a flow adjoint variable as:

$$\mathbf{\Lambda}_u = -\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right]^{-T}\frac{\partial L}{\partial \mathbf{u}}^T \tag{51}$$

and a mesh adjoint variable as:

$$\mathbf{\Lambda}_x = [K]^{-T}\left[\frac{\partial L}{\partial \mathbf{x}}^T + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}^T \mathbf{\Lambda}_u\right] \tag{52}$$

then the complete adjoint problem consists of solving first the flow adjoint problem

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right]^T \mathbf{\Lambda}_u = -\frac{\partial L}{\partial \mathbf{u}}^T \tag{53}$$

and then solving the mesh adjoint problem

$$[K]^T \mathbf{\Lambda}_x = \left[\frac{\partial L}{\partial \mathbf{x}}^T + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}^T \mathbf{\Lambda}_u\right] \tag{54}$$

after the required inner products have been formed to evaluate the right-hand side of this equation. The final objective sensitivity is then obtained as:

$$\frac{dL}{dD}^T = \frac{\partial \mathbf{x}_s}{\partial D}^T \mathbf{\Lambda}_x \tag{55}$$

3.2.2 Application to Time-Dependent Problems

For time-dependent problems, the objective may consist of an output computed at the final time of the simulation, or may be constructed as a time-integrated quantity. We consider the formulation for a time integrated objective, since this corresponds to the most general case. Thus, our general objective can be written as:

$$L^g = \int_0^T L(t)dt \tag{56}$$

When discretized in time, the integral form becomes

$$L^g(D) = \sum_{n=0}^{n=n_f} \omega_n L^n(\mathbf{u}^n(D), \mathbf{x}^n(D)) \tag{57}$$

where ω_n represents the quadrature weight associated with each time step value, and L^n represents the objective evaluated at the time step n, which depends directly only on the grid coordinates and flow variables at the time level n. Using the chain rule, the sensitivity vector of this objective is given as:

$$\frac{dL^g}{dD} = \sum_{n=0}^{n=n_f} \omega_n \left[\frac{\partial L^n}{\partial \mathbf{u}^n} \frac{\partial \mathbf{u}^n}{\partial D} + \frac{\partial L^n}{\partial \mathbf{x}^n} \frac{\partial \mathbf{x}^n}{\partial D} \right] \tag{58}$$

As previously, the flow sensitivities are given by the constraint that the flow equations at each implicit time step must be satisfied. However, in the time-dependent case, the flow residuals depend on values at previous time steps in addition to values at the current time step n. Considering a BDF1 time-integration scheme for simplicity, the flow residuals at time level n can be written as:

$$\mathbf{R}^n(\mathbf{u}^n(D), \mathbf{u}^{n-1}(D), \mathbf{x}^n(D), \mathbf{x}^{n-1}(D)) = 0 \tag{59}$$

which gives the following expression for the flow sensitivities at time level n:

$$\left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n} \right] \frac{\partial \mathbf{u}^n}{\partial D} = - \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n} \frac{\partial \mathbf{x}^n}{\partial D} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^{n-1}} \frac{\partial \mathbf{x}^{n-1}}{\partial D} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}} \frac{\partial \mathbf{u}^{n-1}}{\partial D} \right] \tag{60}$$

Similarly, the mesh sensitivities at time level n can be written as a function of the mesh sensitivities at the previous time level as:

$$\frac{\partial \mathbf{x}^n}{\partial D} = [K^n]^{-1} \frac{\partial \mathbf{x}^{n-1}}{\partial D} \tag{61}$$

where $[K^n]^{-1}$ corresponds to a representation of the mesh motion in going from time level $n-1$ to time level n.

The above equations illustrate how the forward sensitivity or tangent problem reduces to a forward integration in time. Assuming the values of the flow and mesh sensitivities are known at the previous time level $n-1$, the sensitivity values at the new time level n are obtained through the solution of equations (60) and (61), respectively (assuming the flow and mesh states $\mathbf{u}^n$ and $\mathbf{x}^n$ at the new time level have been computed by the analysis problem). Substituting these values into equation (58) and proceeding to the next time step, the process is repeated until the final time level is reached, thus generating the final sensitivities for the time-integrated objective L^g.

If the process is initiated with a prescribed initial flow field, then the initial flow sensitivities at time level $n = 0$ vanish, whereas if the process is initiated from a steady-state solution, the initial flow sensitivities correspond to the steady-state flow sensitivities which are obtained through the solution of equation (47). On the other hand, the mesh sensitivities at the initial time level must be computed through the solution of the mesh deformation equations at $n = 0$ (e.g. $[K^o]$) even in the absence of any time-dependent

motion, since the initial mesh will vary with the values of the design variables which define the shape of the geometry. Additionally, the mesh motion between any two consecutive time steps can consist either of a mesh deformation problem, or a solid body mesh transformation, or both. In the former case, the (iterative) solution of the mesh motion equations is required at each new time step, whereas in the latter case, the $[K^n]^{-1}$ matrix is defined explicitly by the translation/rotation matrices used to displace the mesh.

In order to formulate the time-dependent adjoint problem, the expressions given by equation (60) for the flow sensitivities are substituted into equation (58), and the entire equation is transposed. Considering, for the moment, only the contributions due to the objective value L^n at level n, we obtain:

$$\frac{\partial L^n}{\partial D} = \frac{\partial \mathbf{x}^n}{\partial D}^T \frac{\partial L^n}{\partial \mathbf{x}^n}^T - \left[\frac{\partial \mathbf{x}^n}{\partial D}^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}^T + \frac{\partial \mathbf{x}^{n-1}}{\partial D}^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^{n-1}}^T + \frac{\partial \mathbf{u}^{n-1}}{\partial D}^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}}^T \right] \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n} \right]^T \frac{\partial L^n}{\partial \mathbf{u}^n}^T \quad (62)$$

If we now define a flow adjoint variable at time level n as:

$$\mathbf{\Lambda}_u^n = - \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n} \right]^{-T} \frac{\partial L^n}{\partial \mathbf{u}^n}^T \quad (63)$$

equation (62) can be rearranged as:

$$\frac{\partial L^n}{\partial D} = \frac{\partial \mathbf{x}^{n-1}}{\partial D}^T [K^n]^{-T} \left[\frac{\partial L^n}{\partial \mathbf{x}^n}^T + \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}^T \mathbf{\Lambda}_\mathbf{u}^\mathbf{n} \right] + \frac{\partial \mathbf{x}^{n-1}}{\partial D}^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^{n-1}}^T \mathbf{\Lambda}_\mathbf{u}^\mathbf{n} + \frac{\partial \mathbf{u}^{n-1}}{\partial D}^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}}^T \mathbf{\Lambda}_\mathbf{u}^\mathbf{n} \quad (64)$$

where we have also substituted in the expression for $\frac{\partial x^n}{\partial D}$ using equation (61). The first term on the right-hand side of this equation is similar in form to the steady-state sensitivities derived in equation (50). We next define a mesh adjoint variable at time level n as:

$$\mathbf{\Lambda}_\mathbf{x}{}^n = [K^n]^{-T} \left[\frac{\partial L^n}{\partial \mathbf{x}^n}^T + \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}^T \mathbf{\Lambda}_\mathbf{u}^\mathbf{n} \right] \quad (65)$$

and note that this quantity is entirely computable at the end of the analysis run, given the solution values at time level n. When substituted into equation (64), the remaining terms depend only on values at previous time steps, thus leading to a backwards recurrence relation in time. In order to obtain expressions for the remaining terms, we substitute into this equation the constraint equation for the flow sensitivities at time level $n-1$, which is similar to equation (60), although with different time level indices, i.e.

$$\left[\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-1}} \right] \frac{\partial \mathbf{u}^{n-1}}{\partial D} = - \left[\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{x}^{n-1}} \frac{\partial \mathbf{x}^{n-1}}{\partial D} + \frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{x}^{n-2}} \frac{\partial \mathbf{x}^{n-2}}{\partial D} + \frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-2}} \frac{\partial \mathbf{u}^{n-2}}{\partial D} \right] \quad (66)$$

The procedure then consists of factorizing all terms multiplying $\frac{\partial \mathbf{x}^{n-1}}{\partial D}$ and premultiplying these by the mesh adjoint operator $[K^{n-1}]^T$ to obtain expressions in terms of the next preceding time level $n-2$.

However, for the full time-integrated objective given in equation (57), the contributions from the L^{n-1} sensitivities at the $n-1$ time level must also be considered, namely, by analogy with equation (62):

$$\frac{\partial L^{n-1}}{\partial D} = \frac{\partial \mathbf{x}^{n-1}}{\partial D}^T \frac{\partial L^{n-1}}{\partial \mathbf{x}^{n-1}}^T - \left[\frac{\partial \mathbf{x}^{n-1}}{\partial D}^T \frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{x}^{n-1}}^T \right] \left[\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-1}} \right]^T \frac{\partial L^{n-1}}{\partial \mathbf{u}^{n-1}}^T + ... \quad (67)$$

When all these terms are substituted into equation (58) and factorized appropriately, the resulting expression can be written as:

$$\frac{dL^g}{dD} = \frac{\partial \mathbf{x}^{n-2\,T}}{\partial D}\ \mathbf{\Lambda}_x^{n-1} + ...\text{previous time step terms...} \tag{68}$$

with

$$\mathbf{\Lambda}_x^{n-1} = \left[K^{n-1}\right]^{-T}\left[\mathbf{\Lambda}_x^n + \frac{\partial \mathbf{R}^{n\ T}}{\partial \mathbf{x}^{n-1}}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{n} + \omega_{n-1}\frac{\partial L^{n-1\,T}}{\partial \mathbf{x}^{n-1}} + \frac{\partial \mathbf{R}^{n-1\,T}}{\partial \mathbf{x}^{n-1}}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{n-1}\right] \tag{69}$$

and

$$\mathbf{\Lambda}_\mathbf{u}^\mathbf{n-1} = \left[\frac{\partial \mathbf{R}^{n-1}}{\partial \mathbf{u}^{n-1}}\right]^{-T}\left[\omega_{n-1}\frac{\partial L^{n-1\,T}}{\partial \mathbf{u}^{n-1}} + \frac{\partial \mathbf{R}^{n\ T}}{\partial \mathbf{u}^{n-1}}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{n}\right] \tag{70}$$

where the $\omega_n, \omega_{n-1}, ...$ quadrature weights are included in order to obtain the global time-integrated sensitivity as per equation (58). When equations (70) and (69) are solved and substituted into equation (68), an expression depending only on $n-2$ and earlier time levels is obtained, and the entire process may then be repeated to advance to the next earlier time level. Thus, the unsteady adjoint sensitivity calculation corresponds to a backwards integration in time, and requires the solution of one flow adjoint and one mesh motion adjoint problem at each time step. In this sense, the time-dependent adjoint problem at each time step is similar to a steady-state adjoint problem, although the right-hand side of the unsteady flow and mesh adjoint problems contain additional terms. For example, at a given time level $n = k$, the flow and mesh adjoint problems can be written as:

$$\mathbf{\Lambda}_\mathbf{u}^\mathbf{k} = \left[\frac{\partial \mathbf{R}^k}{\partial \mathbf{u}^k}\right]^{-T}\left[\omega_k\frac{\partial L^{k\,T}}{\partial \mathbf{u}^k} + \left(\frac{\partial \mathbf{R}^{k+1\,T}}{\partial \mathbf{u}^k}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{k+1}\right)\right] \tag{71}$$

and

$$\mathbf{\Lambda}_x^k = \left[K^k\right]^{-T}\left[+\omega_k\frac{\partial L^{k\,T}}{\partial \mathbf{x}^k} + \frac{\partial \mathbf{R}^{k\,T}}{\partial \mathbf{x}^k}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{k} + \left(\mathbf{\Lambda}_x^{k+1} + \frac{\partial \mathbf{R}^{k+1\,T}}{\partial \mathbf{x}^k}\ \mathbf{\Lambda}_\mathbf{u}^\mathbf{k+1}\right)\right] \tag{72}$$

respectively, where the bracketed expressions correspond to the additional terms not present in the steady flow and mesh adjoint equations, and imply a recurrence relation from later to earlier time levels. Note also that at the beginning of the adjoint time-integration (i.e. at the final time step $k = n_f$), these terms vanish since no later time step values $k+1$ exist, and the adjoint problems take on a form similar to the steady-state flow and mesh adjoint problems at the final simulation time step, as given in equation (63).

3.2.3 Additional Term Formulation

As can be seen from the above analysis, the unsteady adjoint problem may be implemented as a conceptually straight-forward extension of the steady-state adjoint problem,

replicating the steady adjoint flow and mesh problems as well as solution strategies at each time level, with additional source terms as outlined above. These additional source terms are constructed from linearization terms which are only present in the unsteady residual, i.e. $\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}}$ and $\frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^{n-1}}$. However, the other terms previously present in the steady-state formulation, i.e. $\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n}$ and $\frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}$ must also be modified to take into account the different form of the unsteady residual, as given by equation (38). In both instances, terms arising from the linearization of the residual with respect to the flow variables are relatively straight-forward. For example, the Jacobian $\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n}$ must be augmented by the diagonal term $\frac{[I]\mathbf{V}^n}{\Delta t}$, due to the linearization of the first term in the time discretization of equation (38). Similarly, the linearization of the unsteady residual with respect to previous time level flow variables, for a BDF1 scheme is seen to be:

$$\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^{n-1}} = [I]\,\frac{\mathbf{V}^{n-1}}{\Delta t} \tag{73}$$

In the context of the adjoint formulation, the matrix-vector product involving this term in equation (70) simply results in a rescaling of the adjoint vector $\mathbf{\Lambda}_u^n$ by the scalar value $\frac{\mathbf{V}^{n-1}}{\Delta t}$.

On the other hand, the modifications required for the linearization of the residual with respect to the mesh coordinates are more involved. As in the steady-state case, the term $\frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}$ must take into account the direct effect of coordinate changes on the residual, as well as changes in all mesh metrics due to changes in the mesh coordinates at the current time level. However, for the unsteady case, the face integrated grid speed terms $\dot{\mathbf{x}}$ also depend on the current grid coordinate values, and must be linearized accordingly. Additionally, the leading term of the BDF time discretization depends on the control volume values computed at the current time level $\mathbf{V}(x^n)$. In general, $\frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}$ constitutes a sparse rectangular matrix which can have a fairly complicated structure. However, since only the product of this matrix with a field vector is required in the tangent or adjoint model, the forward matrix-vector product may be evaluated as:

$$\frac{d\mathbf{R}^n}{d\mathbf{x}^n}.\delta\mathbf{x}^n = \frac{\partial \mathbf{R}^n}{\partial \mathbf{V}^n}.\frac{\partial \mathbf{V}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n + \frac{\partial \mathbf{R}^n}{\partial \dot{\mathbf{x}}^n}.\frac{\partial \dot{\mathbf{x}}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n + \frac{\partial \mathbf{R}^n}{\partial \mathbf{fn}^n}.\frac{\partial \mathbf{fn}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n + \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n \tag{74}$$

where $\delta\mathbf{x}^n$ represents the input vector, $\mathbf{fn}^n$ represents the vector of control volume face normals over the mesh (which are constructed using only the current time level mesh coordinates for BDF schemes), $\mathbf{V}^n$ represents the vector of mesh control volumes, and $\dot{\mathbf{x}}^n$ represents the face-integrated grid velocities, which depend both on current and previous time levels. The evaluation of this matrix-vector product is performed in a multi-step procedure, given as

$$\delta\mathbf{V}^n = \frac{\partial \mathbf{V}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n \tag{75}$$

$$\delta\dot{\mathbf{x}}^n = \frac{\partial \dot{\mathbf{x}}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n \tag{76}$$

$$\delta\mathbf{fn}^n = \frac{\partial \mathbf{fn}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n \tag{77}$$

$$\frac{d\mathbf{R}^n}{d\mathbf{x}^n}\delta\mathbf{x} = \frac{\partial \mathbf{R}^n}{\partial \mathbf{V}^n}\delta\mathbf{V}^n + \frac{\partial \mathbf{R}^n}{\partial \mathbf{fn}^n}\delta\mathbf{fn}^n + \frac{\partial \mathbf{R}^n}{\partial \dot{\mathbf{x}}^n}\delta\dot{\mathbf{x}}^{\mathbf{n}} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{x}^n}\delta\mathbf{x}^n \tag{78}$$

Note that the first step involves a matrix-vector product based on the linearization of the volume terms, while the second step involves the linearization of the the face-integrated grid velocity terms, and the third step is based on the linearization of the mesh metric routines. Finally, the last step involves only the linearization of the flow residual with respect to terms which appear directly in the residual routine (i.e. volumes, grid velocities, mesh metrics, and direct dependence on coordinates). In the case of the adjoint model, the matrix-vector product of equation (74) can be evaluated in a similar fashion. Taking the transpose of equation (74), we obtain the relation:

$$\frac{d\mathbf{R}^{n\,T}}{d\mathbf{x}^n}\ \Lambda_u^n = \frac{\partial \mathbf{V}^{n\,T}}{\partial \mathbf{x}^n}\ \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{V}^n}\ \Lambda_u^n + \frac{\partial \dot{\mathbf{x}}^{n\,T}}{\partial \mathbf{x}^n}\ \frac{\partial \mathbf{R}^{n\,T}}{\partial \dot{\mathbf{x}}^n}\ \Lambda_u^n + \frac{\partial \mathbf{fn}^{n\,T}}{\partial \mathbf{x}^n}\ \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{fn}^n}\ \Lambda_u^n + \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{x}^n}\ \Lambda_u^n \tag{79}$$

which can be evaluated in a multi-step procedure as:

$$\delta \mathbf{V}^n = \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{V}^n}\ \Lambda_u^n \tag{80}$$

$$\delta \dot{\mathbf{x}}^n = \frac{\partial \mathbf{R}^{n\,T}}{\partial \dot{\mathbf{x}}^n}\ \Lambda_u^n \tag{81}$$

$$\delta \mathbf{fn}^n = \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{fn}^n}\ \Lambda_u^n \tag{82}$$

$$\frac{d\mathbf{R}^{n\,T}}{d\mathbf{x}^n}\ \Lambda_u^n = \frac{\partial \mathbf{V}^{n\,T}}{\partial \mathbf{x}^n}\ \delta \mathbf{V}^n + \frac{\partial \dot{\mathbf{x}}^{n\,T}}{\partial \mathbf{x}^n}\ \delta \dot{\mathbf{x}}^n + \frac{\partial \mathbf{fn}^{n\,T}}{\partial \mathbf{x}^n}\ \delta \mathbf{fn}^n + \frac{\partial \mathbf{R}^{n\,T}}{\partial \mathbf{x}^n}\ \Lambda_u^n \tag{83}$$

As in the previous case, a (transposed) linearization of each individual routine is invoked, but in the reverse order of that used by the original discretization in the tangent model. For example, the first three steps involve the linearization of the flow residual with respect to terms appearing directly in the residual construction (i.e. volumes, grid velocities and grid metrics), while the last step incorporates the linearization of each of these individual terms with respect to the grid coordinates.

Similarly, evaluation of the matrix-vector product involving the linearization of the residual with respect to previous time level grid coordinates is performed for the forward problem as:

$$\frac{d\mathbf{R}^n}{d\mathbf{x}^{n-1}}.\delta \mathbf{x}^{n-1} = \frac{\partial \mathbf{R}^n}{\partial \mathbf{V}^{n-1}}.\frac{\partial \mathbf{V}^{n-1}}{\partial \mathbf{x}^{n-1}}\delta \mathbf{x}^{n-1} + \frac{\partial \mathbf{R}^n}{\partial \dot{\mathbf{x}}^n}.\frac{\partial \dot{\mathbf{x}}^n}{\partial \mathbf{x}^{n-1}}\delta \mathbf{x}^{n-1} \tag{84}$$

and for the adjoint problem as:

$$\frac{d\mathbf{R}^{n\ T}}{d\mathbf{x}^{n-1}}\ \Lambda_u^n = \frac{\partial \mathbf{V}^{n-1\,T}}{\partial \mathbf{x}^{n-1}}\ \frac{\partial \mathbf{R}^{n\ T}}{\partial \mathbf{V}^{n-1}}\ \Lambda_u^n + \frac{\partial \dot{\mathbf{x}}^{n\ T}}{\partial \mathbf{x}^{n-1}}\ \frac{\partial \mathbf{R}^{n\,T}}{\partial \dot{\mathbf{x}}^n}\ \Lambda_u^n \tag{85}$$

where it is seen that the only dependence on previous time levels (in this case for the BDF1 scheme) occurs through the volume term evaluated at the previous time level arising from the second contribution to the time discretization, and the face-integrated grid velocity terms present in the spatial residual.

3.3 Unsteady Pitching Wing Optimization Example

A three-dimensional pitching wing is employed as a test problem for demonstrating the unsteady adjoint formulation within the context of a time-dependent shape optimization

problem. The geometry consists of an ONERA M6 wing which oscillates about a non-swept spanwise axis which intersects the symmetry plane at the quarter-chord of the root section. The pitching motion is sinusoidal with an amplitude of 25^o about a mean incidence of 5^o, and a reduced frequency of 0.1682, and the freestream Mach number is 0.3. An unstructured mesh of approximately 100,000 points containing prismatic elements in the boundary layer region and tetrahedral elements in inviscid flow regions is used to discretize the computational domain. The time-dependent mesh motion is determined by rotating the entire mesh as a solid body at each time step, in response to the prescribed wing motion. The initial condition consists of a precomputed steady-state flow-field for the wing at 5^o incidence. The unsteady Reynolds-averaged Navier-Stokes equations are discretized in finite-volume ALE form, using the Spalart-Allmaras turbulence model [33], and solved using an agglomeration multigrid strategy at each implicit time step [14, 19]. A total of 20 time steps are used to advance the solution for one period of the pitching motion. Figure 1 illustrates the ONERA M6 wing and unstructured mesh used for the calculations. In Figure 3, the time history of the computed lift and drag coefficients as a function of the pitching incidence are given.

3.3.1 Unsteady Objective Function Formulation

An approximation to a time-integrated objective function, as defined previously by equation (57) is used for this test case. The objective function is based on the summation of the differences between a target and a computed objective value at each time level n. Figure 2 exemplifies the formulation of this unsteady objective function. If C_L^n and C_D^n refer to the lift and drag coefficients computed on the wing at time-level n during the pitch cycle, then the local objective at time-level n is defined as:

$$L^n = (\delta C_L^n)^2 + 10(\delta C_D^n)^2 \tag{86}$$

$$\delta C_L^n = (C_L^n - C_{Ltarget}^n) \tag{87}$$

$$\delta C_D^n = (C_D^n - C_{Dtarget}^n) \tag{88}$$

where the factor of 10 for the drag coefficient is introduced to equalize its order of magnitude with the lift coefficient. The global or time-integrated objective is taken to be the root-mean-square (RMS) average of the local objective functions:

$$L^g = \sqrt{\frac{\sum_{n=0}^{n=n_f} L^n}{n_f + 1}} \tag{89}$$

As the value of the objective function L^g is minimized, the computed time-dependent force coefficients are driven towards the target distributions.

3.3.2 Time Dependent Optimization

The unsteady adjoint formulation is used to demonstrate a time-dependent optimization problem for the pitching ONERA M6 wing problem. The target time-dependent lift and drag profiles depicted in Figure 3 are prescribed and used to construct the objective function defined by equation (89). In order to ensure these profiles are realizable, they have been constructed first by deforming the original ONERA M6 wing surface, and

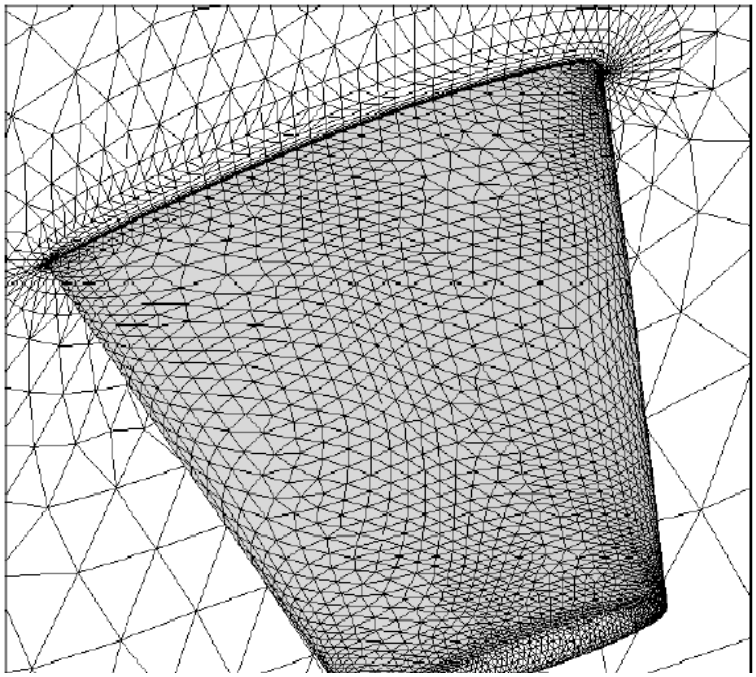
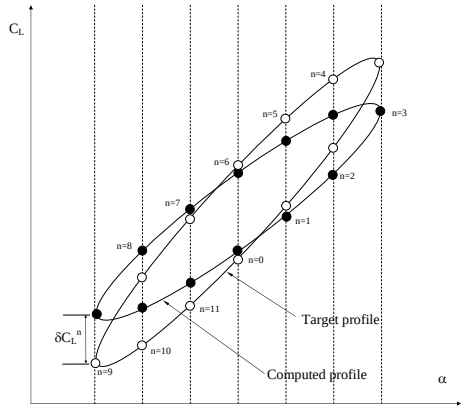
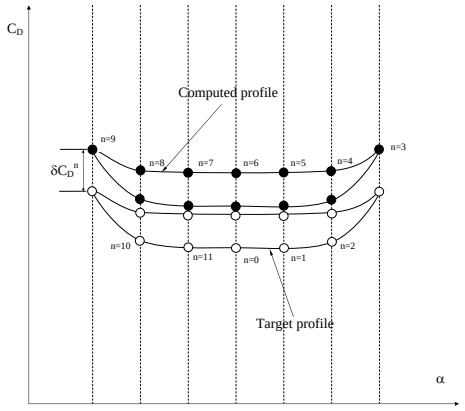


Figure 1: Illustration mixed prismatic-tetrahedral unstructured mesh used for calculation of pitching ONERA M6 wing.



2.1: Time dependent lift profile



2.2: Time dependent drag profile

Figure 2: Illustration of time-dependent objective function constructed as difference between prescribed and computed time-dependent lift and drag profiles.

computing the pitching wing problem with the deformed configuration. The optimization problem thus consists of recovering this deformed configuration, starting from the initial ONERA M6 wing geometry. For this simple case, the design variables are defined as the normal displacement values assigned to the individual wing surface grid points.

The optimization procedure follows the steepest descent approach described by Jameson [8, 7]. Once the objective function sensitivities $\frac{dL}{dD}$ have been computed using the method described above, an increment in the design variables is prescribed as:

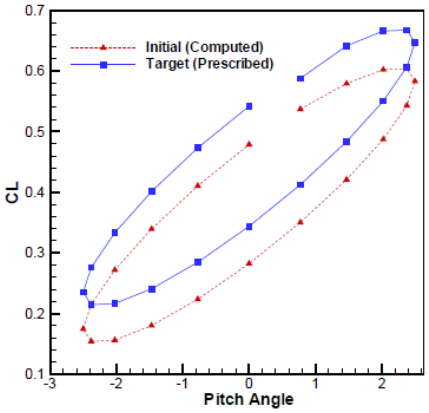
$$\delta D = -\lambda \frac{\tilde{dL}}{dD} \tag{90}$$

where λ represents a step size chosen small enough to ensure convergence of the optimization procedure, and $\frac{\tilde{dL}}{dD}$ represents the smoothed gradients $\frac{dL}{dD}$, obtained using an implicit smoothing technique, which is necessary to ensure smooth design shapes, as described in reference [8]. The implementation of the steepest descent optimization procedure in this example is not optimal, in that λ is determined empirically, but is sufficient for demonstrating the utility of the adjoint solution techniques described herein.

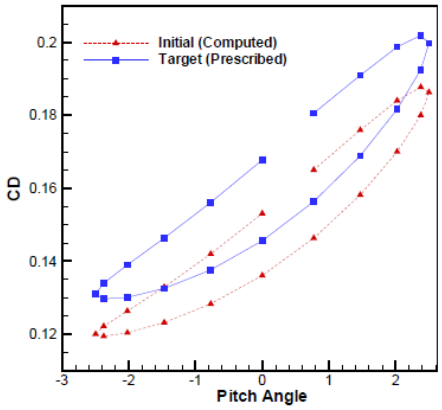
A total of 25 design cycles are used for the optimization problem. At each design cycle, an unsteady problem consisting of 20 implicit time steps is first solved, followed by the solution of the adjoint problem proceeding backwards in time. At each time level in the analysis procedure, the entire flow field and set of grid coordinates are written out, in partitioned format, to a binary file on the local disk on each node of the parallel computer cluster. During the adjoint solution procedure, the partitioned solution and grid coordinate files on each node are read back in by the code at each time step, prior to the calculation of the sensitivity contributions at the given time step. Within the optimization run, the output and input of these solution files is essentially invisible to the user, since they occur at each design cycle and overwrite the previous files produced at earlier design iterations.

Figures 4 depict the lift and drag time histories for the final optimized wing configuration, showing how these match closely with the target distributions, and indicating that an optimum configuration has been achieved for this design problem. In Figure 5 the optimization procedure is seen to result in a reduction of the time-integrated objective function by approximately three orders of magnitude over 25 design cycles.

This basic optimization problem is relatively simple in that it is unconstrained, and makes use of a relatively simple geometry with surface mesh point design variables and employs a basic steepest-descent optimizer. However, this example serves the purpose of illustrating how the discrete adjoint formulation described in detail in this section can be implemented and used for a time-dependent shape optimization problem, while avoiding the introduction of additional complications. In the following sections we extend this formulation to more complex multidisciplinary time-dependent problems, and provide a demonstration using a more realistic shape optimization problem using more sophisticated design parameters, objectives, constraints, and optimizer. The additional topic of verification of the discrete sensitivities is also deferred to the next section.

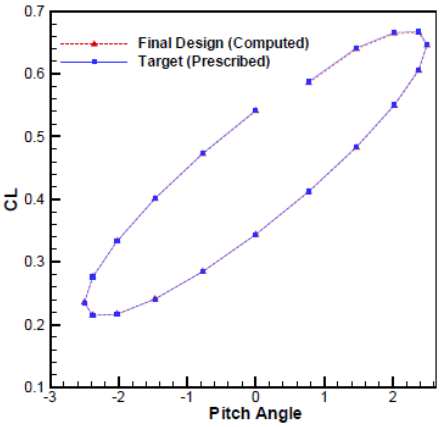


3.1:

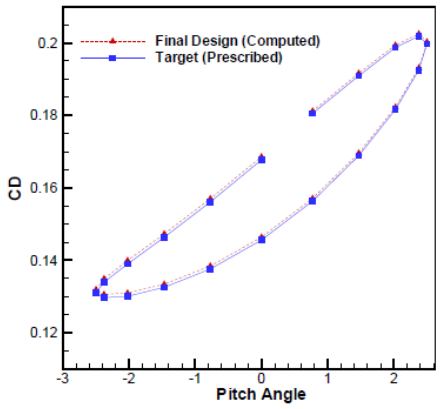


3.2:

Figure 3: Initial and target lift and drag time histories for pitching wing optimization problem.

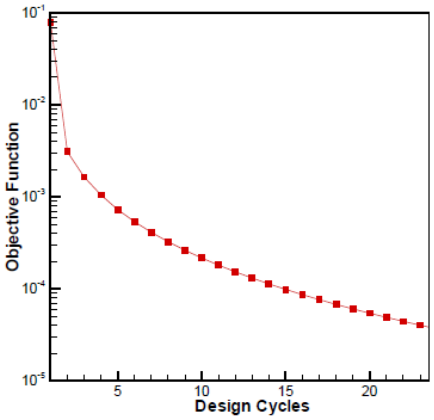


4.1:



4.2:

Figure 4: Optimized and target lift and drag time histories for pitching wing optimization problem.



5.1:

Figure 5: Convergence of objective function with design iterations for pitching wing optimization problem.

4 Generalized Gradient Formulation for Time-Dependent Multidisciplinary Coupled Equations

In the previous section a detailed derivation of the discrete tangent and adjoint sensitivity formulations was given for time-dependent aerodynamic problems. In this section we extend this derivation to time-dependent multidisciplinary problems. As the derivation in the previous section is fairly detailed, pursuing a straight-forward extension of this approach to multidisciplinary time-dependent problems results in substantially increased complexity. Therefore, in order to avoid getting lost in algebraic complexity and notation, we adopt a higher level notation for describing the governing equations, constraints and objective functionals for multidisciplinary problems. From the outset, we do not specify any particular functional forms and assume all disciplines are fully coupled in the most general formulation. For each discipline, we denote the equations to be solved as a single residual equation that spans all space and time and take inner products over space and time. While this higher level generic notation hides many of the implementation details, it conveys the overall strategy of deriving and solving multidisciplinary sensitivity analysis problems and considerably simplifies the notation. In all cases, details at the individual disciplinary levels can be recovered by reverting to the derivation described in the previous section.

4.1 Generalized Forward Linearization

We consider an arbitrary number of m coupled unsteady disciplines, where the solution of the n^{th} discipline is given by the state variable $\mathscr{U}_n$. The state variables for each of the disciplines span both the spatial and temporal domains since it is assumed that the governing equations of every discipline are unsteady in nature. For the general case, a scalar functional L evaluated using the multidisciplinary solution set can be expressed as

$$L = L(D, \mathscr{U}_1, \mathscr{U}_2, \cdots, \mathscr{U}_m) \tag{91}$$

where in addition to an explicit dependence on the design inputs D, there exists an implicit dependence through the state variables $\boldsymbol{\mathcal{U}}$ of each discipline. Differentiating with respect to the design inputs using the chain rule, the total sensitivity or gradient vector $\frac{dL}{dD}$ can be expressed as

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_1}\frac{\partial \boldsymbol{\mathcal{U}}_1}{\partial D} + \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_2}\frac{\partial \boldsymbol{\mathcal{U}}_2}{\partial D} + \cdots + \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_m}\frac{\partial \boldsymbol{\mathcal{U}}_m}{\partial D} \tag{92}$$

which can also be expressed in terms of the inner product between the vector of functional sensitivities to state variables and the vector of state variable sensitivities to the design inputs as

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_1} & \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_2} & \cdots & \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_m} \end{bmatrix} \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{U}}_1}{\partial D} \\ \frac{\partial \boldsymbol{\mathcal{U}}_2}{\partial D} \\ \vdots \\ \frac{\partial \boldsymbol{\mathcal{U}}_m}{\partial D} \end{bmatrix} \tag{93}$$

Now assume that the spatial domain is represented by n_{cells} discrete elements while the temporal domain consists of n_{steps} discrete time-steps. The linearization of the functional with respect to the disciplinary state variables (i.e $\frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_i}$) then forms vectors of dimension $[1 \times n_{steps}]$, where each element in the vector is a vector by itself of dimension $[1 \times n_{cells}]$. These vectors are easily computable since the functional L is a scalar quantity. The linearization of the disciplinary state variables with respect to the design inputs (i.e. $\frac{\partial \boldsymbol{\mathcal{U}}_i}{\partial D}$) form matrices of dimension $[n_{ncells} \times n_D]$ at each time-step for the case of n_D design inputs. For each discipline, there are n_{steps} number of such matrices. The state variable sensitivity matrices are unknown quantities at this point and an expression for their evaluation must be determined. The governing nonlinear equations of each discipline may be cast in residual form as

$$\begin{aligned} \boldsymbol{\mathcal{R}}_1(D, \boldsymbol{\mathcal{U}}_1, \boldsymbol{\mathcal{U}}_2, \cdots, \boldsymbol{\mathcal{U}}_m) &= 0 \\ \boldsymbol{\mathcal{R}}_2(D, \boldsymbol{\mathcal{U}}_1, \boldsymbol{\mathcal{U}}_2, \cdots, \boldsymbol{\mathcal{U}}_m) &= 0 \\ &\vdots \\ \boldsymbol{\mathcal{R}}_m(D, \boldsymbol{\mathcal{U}}_1, \boldsymbol{\mathcal{U}}_2, \cdots, \boldsymbol{\mathcal{U}}_m) &= 0 \end{aligned} \tag{94}$$

where it is important to note that the residual equation of each discipline is dependent not only on its own state but also on the states of the remaining disciplines due to the assumption of coupling between disciplines. For the general case, the residual equations also have explicit dependence on the design inputs D and an implicit dependence through the state variables $\boldsymbol{\mathcal{U}}$. For a fully converged solution set $\boldsymbol{\mathcal{U}}$, the residual equations must evaluate identically to zero at all spatial and temporal points in the domain, which constitutes the constraints to be satisfied. Differentiating the disciplinary residual equations

(constraints) with respect to the set of design inputs D then yields:

$$\begin{aligned}
&\frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_1}\frac{\partial \boldsymbol{\mathscr{U}}_1}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_2}\frac{\partial \boldsymbol{\mathscr{U}}_2}{\partial D} + \cdots + \frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_m}\frac{\partial \boldsymbol{\mathscr{U}}_m}{\partial D} = 0 \\
&\frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_1}\frac{\partial \boldsymbol{\mathscr{U}}_1}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_2}\frac{\partial \boldsymbol{\mathscr{U}}_2}{\partial D} + \cdots + \frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_m}\frac{\partial \boldsymbol{\mathscr{U}}_m}{\partial D} = 0 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_1}\frac{\partial \boldsymbol{\mathscr{U}}_1}{\partial D} + \frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_2}\frac{\partial \boldsymbol{\mathscr{U}}_2}{\partial D} + \cdots + \frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_m}\frac{\partial \boldsymbol{\mathscr{U}}_m}{\partial D} = 0
\end{aligned} \tag{95}$$

which can further be combined into matrix form by extracting the vector of unknown state variable sensitivity matrices as:

$$\begin{bmatrix}
\frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_1} & \frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_2} & \cdots & \frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial \boldsymbol{\mathscr{U}}_m} \\
\frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_1} & \frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_2} & \cdots & \frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial \boldsymbol{\mathscr{U}}_m} \\
\vdots & & & \\
\frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_1} & \frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_2} & \cdots & \frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial \boldsymbol{\mathscr{U}}_m}
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \boldsymbol{\mathscr{U}}_1}{\partial D} \\
\frac{\partial \boldsymbol{\mathscr{U}}_2}{\partial D} \\
\vdots \\
\frac{\partial \boldsymbol{\mathscr{U}}_m}{\partial D}
\end{bmatrix} =
\begin{bmatrix}
-\frac{\partial \boldsymbol{\mathscr{R}}_1}{\partial D} \\
-\frac{\partial \boldsymbol{\mathscr{R}}_2}{\partial D} \\
\vdots \\
-\frac{\partial \boldsymbol{\mathscr{R}}_m}{\partial D}
\end{bmatrix} \tag{96}$$

The Jacobian matrices $\frac{\partial \boldsymbol{\mathscr{R}}_i}{\partial \boldsymbol{\mathscr{U}}_j}(i = 1, m : j = 1, m)$ span both spatial and temporal domains, however time being purely hyperbolic in nature implies that the discrete residual $\boldsymbol{\mathscr{R}}^n$ at an arbitrary time-step n can only depend on quantities at time indices $\leq n$. Consequently, for the general case, any Jacobian matrix when discretely expanded in time for a temporal domain consisting of n time-steps is block lower triangular as shown below, where each of the blocks is a matrix spanning the spatial domain.

$$\frac{\partial \boldsymbol{\mathscr{R}}}{\partial \boldsymbol{\mathscr{U}}} =
\begin{bmatrix}
\ddots & 0 & 0 & 0 \\
\cdots & \frac{\partial \boldsymbol{\mathscr{R}}^{n-2}}{\partial \boldsymbol{\mathscr{U}}^{n-2}} & 0 & 0 \\
\cdots & \frac{\partial \boldsymbol{\mathscr{R}}^{n-1}}{\partial \boldsymbol{\mathscr{U}}^{n-2}} & \frac{\partial \boldsymbol{\mathscr{R}}^{n-1}}{\partial \boldsymbol{\mathscr{U}}^{n-1}} & 0 \\
\cdots & \frac{\partial \boldsymbol{\mathscr{R}}^{n}}{\partial \boldsymbol{\mathscr{U}}^{n-2}} & \frac{\partial \boldsymbol{\mathscr{R}}^{n}}{\partial \boldsymbol{\mathscr{U}}^{n-1}} & \frac{\partial \boldsymbol{\mathscr{R}}^{n}}{\partial \boldsymbol{\mathscr{U}}^{n}}
\end{bmatrix} \tag{97}$$

As an example, for two coupled disciplines ($m = 1, 2$) and a temporal domain consisting of two time-steps n and $n-1$, equation (96) can be rewritten using the discrete temporally

expanded form of the Jacobian matrices as

$$\left[\begin{array}{cc|cc} \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_1^{n-1}} & 0 & \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_2^{n-1}} & 0 \\ \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n-1}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n}} \\ \hline \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_1^{n-1}} & 0 & \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_2^{n-1}} & 0 \\ \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n-1}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n}} \end{array}\right] \left[\begin{array}{c} \frac{\partial \mathscr{U}_1^{n-1}}{\partial D} \\ \frac{\partial \mathscr{U}_1^{n}}{\partial D} \\ \hline \frac{\partial \mathscr{U}_2^{n-1}}{\partial D} \\ \frac{\partial \mathscr{U}_2^{n}}{\partial D} \end{array}\right] = \left[\begin{array}{c} -\frac{\partial \mathscr{R}_1^{n-1}}{\partial D} \\ -\frac{\partial \mathscr{R}_1^{n}}{\partial D} \\ \hline -\frac{\partial \mathscr{R}_2^{n-1}}{\partial D} \\ -\frac{\partial \mathscr{R}_2^{n}}{\partial D} \end{array}\right] \tag{98}$$

Swapping rows and columns, the overall system may be rearranged as a block lower triangular matrix of the form shown below, where each diagonal block represents a single time-step.

$$\left[\begin{array}{cc|cc} \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_2^{n-1}} & 0 & 0 \\ \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_2^{n-1}} & 0 & 0 \\ \hline \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n-1}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n}} & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n}} \\ \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n-1}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n-1}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n}} & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n}} \end{array}\right] \left[\begin{array}{c} \frac{\partial \mathscr{U}_1^{n-1}}{\partial D} \\ \frac{\partial \mathscr{U}_2^{n-1}}{\partial D} \\ \hline \frac{\partial \mathscr{U}_1^{n}}{\partial D} \\ \frac{\partial \mathscr{U}_2^{n}}{\partial D} \end{array}\right] = \left[\begin{array}{c} -\frac{\partial \mathscr{R}_1^{n-1}}{\partial D} \\ -\frac{\partial \mathscr{R}_2^{n-1}}{\partial D} \\ \hline -\frac{\partial \mathscr{R}_1^{n}}{\partial D} \\ -\frac{\partial \mathscr{R}_2^{n}}{\partial D} \end{array}\right] \tag{99}$$

For the limiting case of a single design input D, this forms a block lower triangular linear system where the unknown state variable sensitivities and the right-hand-side form vectors rather than matrices. The system can be solved using forward substitution, i.e. a forward sweep in time beginning at time-step $n-1$ and progressing to time-step n. At each time-step, a fully coupled linear system is solved iteratively to obtain the sensitivity of the state variables at that time-step to the single design input. For the case of multiple design inputs D, the forward sweep in time along with the coupled linear solution at each time-step has to be performed for each input in order to construct the state variable sensitivity matrix for each discipline one column at a time. Once the state variable sensitivity matrices are available, they can substituted into equation (93) where the inner product can be evaluated and the complete gradient vector $\frac{dL}{dD}$ becomes available.

4.2 Generalized Adjoint Linearization

In the case of the forward linearization, the number of solutions of equation (99) that are required is directly dependent on the number of design inputs D. This issue can be circumvented by switching to the adjoint or reverse linearization of the same problem by

transposing equation (93). The transpose of the forward linearization can be expressed as

$$\frac{dL}{dD}^T = \frac{\partial L}{\partial D}^T + \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{U}}_1}{\partial D}^T & \frac{\partial \boldsymbol{\mathcal{U}}_2}{\partial D}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{U}}_m}{\partial D}^T \end{bmatrix} \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_1}^T \\ \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_2}^T \\ \vdots \\ \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_m}^T \end{bmatrix} \tag{100}$$

where the goal now is to avoid explicit computation of the unknown state variable sensitivity matrices. A convenient expression for the vector of state variable sensitivity matrices can be obtained by transposing and rearranging equation (96) as

$$\begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{U}}_1}{\partial D}^T & \frac{\partial \boldsymbol{\mathcal{U}}_2}{\partial D}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{U}}_m}{\partial D}^T \end{bmatrix} = \\ - \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial D}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial D}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial D}^T \end{bmatrix} \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_1}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_1}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_1}^T \\ \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_2}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_2}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_2}^T \\ \vdots & & & \\ \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_m}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_m}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_m}^T \end{bmatrix}^{-1} \tag{101}$$

Substituting into the transposed total sensitivity equation (100) yields,

$$\frac{dL}{dD}^T = \frac{\partial L}{\partial D}^T - \\ \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial D}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial D}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial D}^T \end{bmatrix} \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_1}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_1}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_1}^T \\ \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_2}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_2}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_2}^T \\ \vdots & & & \\ \frac{\partial \boldsymbol{\mathcal{R}}_1}{\partial \boldsymbol{\mathcal{U}}_m}^T & \frac{\partial \boldsymbol{\mathcal{R}}_2}{\partial \boldsymbol{\mathcal{U}}_m}^T & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}_m}{\partial \boldsymbol{\mathcal{U}}_m}^T \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_1}^T \\ \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_2}^T \\ \vdots \\ \frac{\partial L}{\partial \boldsymbol{\mathcal{U}}_m}^T \end{bmatrix} \tag{102}$$

Now defining a vector of adjoint variables per discipline spanning the spatial and temporal domains as

$$\begin{bmatrix} \boldsymbol{\Lambda}_{\mathscr{U}_1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2} \\ \vdots \\ \boldsymbol{\Lambda}_{\mathscr{U}_m} \end{bmatrix} = - \begin{bmatrix} \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_1}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_1}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_1}^T \\ \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_2}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_2}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_2}^T \\ \vdots & & & \\ \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_m}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_m}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_m}^T \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial L}{\partial \mathscr{U}_1}^T \\ \frac{\partial L}{\partial \mathscr{U}_2}^T \\ \vdots \\ \frac{\partial L}{\partial \mathscr{U}_m}^T \end{bmatrix} \tag{103}$$

a block coupled linear adjoint system of equations can be recovered as

$$\begin{bmatrix} \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_1}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_1}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_1}^T \\ \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_2}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_2}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_2}^T \\ \vdots & & & \\ \frac{\partial \mathscr{R}_1}{\partial \mathscr{U}_m}^T & \frac{\partial \mathscr{R}_2}{\partial \mathscr{U}_m}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial \mathscr{U}_m}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_{\mathscr{U}_1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2} \\ \vdots \\ \boldsymbol{\Lambda}_{\mathscr{U}_m} \end{bmatrix} = - \begin{bmatrix} \frac{\partial L}{\partial \mathscr{U}_1}^T \\ \frac{\partial L}{\partial \mathscr{U}_2}^T \\ \vdots \\ \frac{\partial L}{\partial \mathscr{U}_m}^T \end{bmatrix} \tag{104}$$

Following the same example presented in the derivation of the forward linearization, for a problem with two disciplines ($m = 2$) and two time-steps n and $n-1$ representing the temporal domain, the adjoint system can be discretely expanded in time and expressed as

$$\left[\begin{array}{cc|cc} \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_2^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n-1}}^T \\ 0 & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n}}^T & 0 & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n}}^T \\ \hline \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_2^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n-1}}^T \\ 0 & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n}}^T & 0 & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n}}^T \end{array}\right] \left[\begin{array}{c} \boldsymbol{\Lambda}_{\mathscr{U}_1}^{n-1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_1}^{n} \\ \hline \boldsymbol{\Lambda}_{\mathscr{U}_2}^{n-1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2}^{n} \end{array}\right] = - \left[\begin{array}{c} \frac{\partial L}{\partial \mathscr{U}_1^{n-1}}^T \\ \frac{\partial L}{\partial \mathscr{U}_1^{n}}^T \\ \hline \frac{\partial L}{\partial \mathscr{U}_2^{n-1}}^T \\ \frac{\partial L}{\partial \mathscr{U}_2^{n}}^T \end{array}\right] \tag{105}$$

which can be rearranged into block upper triangular form as shown below by swapping rows and columns.

$$\left[\begin{array}{cc|cc} \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n-1}}{\partial \mathscr{U}_2^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n-1}}^T \\ \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n-1}}{\partial \mathscr{U}_2^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n-1}}^T & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n-1}}^T \\ \hline 0 & 0 & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_1^{n}}^T & \frac{\partial \mathscr{R}_1^{n}}{\partial \mathscr{U}_2^{n}}^T \\ 0 & 0 & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_1^{n}}^T & \frac{\partial \mathscr{R}_2^{n}}{\partial \mathscr{U}_2^{n}}^T \end{array}\right] \left[\begin{array}{c} \boldsymbol{\Lambda}_{\mathscr{U}_1}^{n-1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2}^{n-1} \\ \hline \boldsymbol{\Lambda}_{\mathscr{U}_1}^{n} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2}^{n} \end{array}\right] = - \left[\begin{array}{c} \frac{\partial L}{\partial \mathscr{U}_1^{n-1}}^T \\ \frac{\partial L}{\partial \mathscr{U}_2^{n-1}}^T \\ \hline \frac{\partial L}{\partial \mathscr{U}_1^{n}}^T \\ \frac{\partial L}{\partial \mathscr{U}_2^{n}}^T \end{array}\right] \tag{106}$$

The system can be solved through back substitution, i.e. a backward sweep in time, where a coupled linear system is solved at time-step n before progressing backward to time-step $n-1$. At each time-step a coupled iterative linear solution is required in order to determine the vector of unknown disciplinary adjoint variables at that time-step. Once the vector of disciplinary adjoint variables spanning the spatial and temporal domains is available, it may be substituted into the total sensitivity equation as

$$\frac{dL}{dD}^T = \frac{\partial L}{\partial D}^T + \left[\begin{array}{cccc} \frac{\partial \mathscr{R}_1}{\partial D}^T & \frac{\partial \mathscr{R}_2}{\partial D}^T & \cdots & \frac{\partial \mathscr{R}_m}{\partial D}^T \end{array}\right] \left[\begin{array}{c} \boldsymbol{\Lambda}_{\mathscr{U}_1} \\ \boldsymbol{\Lambda}_{\mathscr{U}_2} \\ \cdots \\ \boldsymbol{\Lambda}_{\mathscr{U}_m} \end{array}\right] \tag{107}$$

where the effect of the number of design inputs D has been confined to a series of matrix-vector products at the end of the computation. As previously discussed, only a single backward sweep in time with coupled linear solutions at each time step is required in order to compute the coupled adjoint problem of the given objective, independent of the number of design variables. The determination of the gradient vector therefore involves a single forward integration in time to obtain the solution to the coupled unsteady analysis problem, and a single backward sweep in time to obtain the necessary adjoint variables, followed by a series of relatively inexpensive inner products for each design variable as determined by equation (107).

4.3 Aerodynamic Sensitivity Analysis Formulation

Although we have already derived the tangent and adjoint formulations for the aerodynamic shape sensitivity problem in detailed form in the previous section, these derivations are repeated here to demonstrate the succinct form afforded by the new higher level multidisciplianry notation and to illustrate the correspondence of this approach with the previous detailed derivation.

We consider the aerodynamic shape sensitivity problem as a multidisciplinary problem consisting of a mesh deformation discipline and a flow solution discipline. For the mesh

deformation discipline, the associated state variables are the grid coordinates $\mathbf{x}$, whereas the flow field variables $\mathbf{U}$ correspond to the flow solution discipline variables. Thus we can write the residual equations or constraints as:

$$\mathbf{G}(\mathbf{x}, D) = 0 \tag{108}$$
$$\mathbf{R}(\mathbf{U}, \mathbf{x}) = 0 \tag{109}$$

where we have specified only one way coupling between the two disciplines, given the fact that the flow solution depends on the mesh configuration or grid coordinates $\mathbf{x}$, but the mesh configuration is independent of the flow solution and only responds to shape changes due to the design variables. We also note that the design variables do not influence the flow residual directly but only indirectly through their effect on the mesh deformation discipline, and as such are not included in the functional dependence of the flow residual.

Next we consider an arbitrary objective function L that is evaluated using the unsteady flow solution set $\mathbf{U}$ and unsteady mesh solution set $\mathbf{x}$ expressed as:

$$L = L(\mathbf{U}, \mathbf{x}) \tag{110}$$

with no explicit dependence on the design variables D. Assuming that the state variables (i.e.$\mathbf{U}, \mathbf{x}$) are dependent on the design parameters D, the total sensitivity of the objective function L to the set of design inputs can be expressed as the inner product between the vector of state sensitivities to design inputs and the vector of objective sensitivities to the state variables as:

$$\frac{dL}{dD} = \begin{bmatrix} \dfrac{\partial L}{\partial \mathbf{x}} & \dfrac{\partial L}{\partial \mathbf{U}} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathbf{x}}{\partial D} \\ \dfrac{\partial \mathbf{U}}{\partial D} \end{bmatrix} \tag{111}$$

Linearizing the coupled disciplinary constraint equations (e.g. equations (108) and (109)) with respect to the design inputs yields:

$$\begin{bmatrix} \dfrac{\partial \mathbf{G}}{\partial \mathbf{x}} & 0 \\ \dfrac{\partial \mathbf{R}}{\partial \mathbf{x}} & \dfrac{\partial \mathbf{R}}{\partial \mathbf{U}} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathbf{x}}{\partial D} \\ \dfrac{\partial \mathbf{U}}{\partial D} \end{bmatrix} = \begin{bmatrix} -\dfrac{\partial \mathbf{G}}{\partial D} \\ 0 \end{bmatrix} \tag{112}$$

These constitute the forward sensitivity or tangent sensitivity equations. The mesh and flow sensitivity vectors can then be substituted into equation (111) to obtain the complete sensitivity of the objective with respect to the design variable D.

The forward sensitivity approach requires a new solution of equation (112) for each design parameter D. On the other hand, the adjoint approach can obtain the sensitivities for any number of design inputs D at a cost which is independent of the number of design variables. The adjoint problem can be obtained by pre-multiplying equation (112) by the inverse of the large coupled matrix and substituting the resulting expression for the sensitivities into equation (111) and defining adjoint variables as the solution of the

system:

$$\begin{bmatrix} \frac{\partial \mathbf{G}}{\partial \mathbf{x}}^T & \frac{\partial \mathbf{R}}{\partial \mathbf{x}}^T \\ 0 & \frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_x} \\ \mathbf{\Lambda_U} \end{bmatrix} = - \begin{bmatrix} \frac{\partial L}{\partial \mathbf{x}}^T \\ \frac{\partial L}{\partial \mathbf{U}}^T \end{bmatrix} \tag{113}$$

where $\mathbf{\Lambda_U}$ and $\mathbf{\Lambda_x}$ are the flow and mesh adjoint variables respectively. The final objective sensitivities can be obtained as:

$$\frac{dL}{dD}^T = \begin{bmatrix} \frac{\partial \mathbf{G}}{\partial D}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_x} \\ \mathbf{\Lambda_u} \end{bmatrix} \tag{114}$$

Recalling that equation (113) applies over the entire time domain, the back-substitution procedure leads to a reverse integration in time, beginning with the last physical time step and proceeding to the initial time step. Furthermore, the one-way coupling between the flow and mesh disciplines results in a block disciplinary lower triangular matrix form in the tangent approach (e.g. equations (112)) and a block disciplinary upper triangular matrix form in the adjoint approach. Therefore, the tangent problem at each time step can be solved by forward substitution at the disciplinary level, and the adjoint problem can be solved using back-substitution at the disciplinary level, thus reproducing the solution strategies previously outlined in equations (49) and (50) for the tangent and adjoint problems respectively. Thus in addition to being more succinct, the current higher level notation clarifies the disciplinary coupling strategies used in both tangent and adjoint formulations.

4.4 Application to Time-Dependent Aeroelastic Problems

We next extend the tangent and adjoint formulations to coupled time-dependent aeroelastic problems. In addition to the flow solution and mesh deformation disciplines, a structural model and a fluid structure interface must now be included as additional disciplines in the formulation, and the coupling between these different disciplines must be considered carefully in the formulation. We begin with a high level conceptual description of the additional disciplines, and proceed to derive the analysis, tangent and adjoint formulations and solution procedures for the coupled aeroelastic problem based on the approach given in the previous sections.

4.4.1 Structural Model Formulation

We consider the equations for a generic structural dynamics model used to represent typical aerospace structures. These may consist of linear or non-linear full finite-element models using shell and brick elements, or of simpler beam models which are often used for modeling slender wings or blade structures for rotary wing configurations. In previous work we have developed and employed a non-linear bend-twist beam finite-element model with 15 degrees of freedom per beam element for rotorcraft applications, as shown in Figure 6(left). On-going work is concerned with the development and implementation of brick and shell-based finite-element models for use in fixed and rotary wing applications,

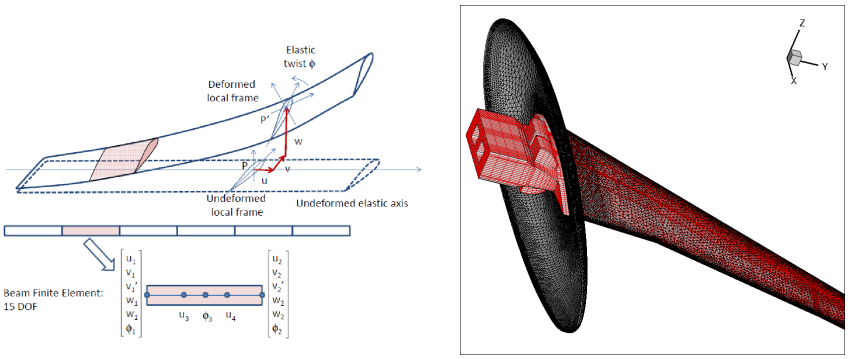


Figure 6: Illustration of structural models for aeroelastic simulations: (left) Finite element beam model with 15 degrees of freedom per element (flap, lag, torsional and axial) used for rotary wing configurations, (right) Brick based finite-element model for flexible wing HIRENASD test case from first Aeroelastic Prediction Workshop (AePW) [29].

as shown in Figure 6(right). In all cases, the second order equation of motion for these structural models can be expressed as:

$$[M]\ddot{\mathbf{q}} + [C]\dot{\mathbf{q}} + [K] = \mathbf{F_b} \tag{115}$$

where $[M]$, $[C]$ and $[K]$ are mass, damping and stiffness matrices of the system of equations assembled through the finite-element discretization. The vector $\mathbf{F_b}$ represents the time-dependent aerodynamic forces applied on the structural model, and the vector $\mathbf{q}$ represents the displacements or degrees of freedom of the structural model. This set of equations can be reduced to a first-order system and solved using a second-order backward difference formula (BDF2) time integration with standard Newton-type linearization and sub-iterations to efficiently invert the implicit system:

$$[I]\dot{\mathbf{Q}} + [A]\mathbf{Q} = \mathbf{F_B} \tag{116}$$

where $[I]$ is the identity matrix, $\mathbf{Q} = [q, \dot{q}]^T$, $\mathbf{F_B} = [0,\ [M]^{-1}F_b]^T$ and $[A] = \begin{bmatrix} 0 & -[I] \\ [M]^{-1}[K] & [M]^{-1}[C] \end{bmatrix}$. The residual of the structural equations can be defined as: $\mathbf{J} = [I]\ \dot{\mathbf{Q}} + [A]\ \mathbf{Q} - \mathbf{F_B} = 0$, and can be expressed in a simplified form as:

$$\mathbf{J}(\mathbf{Q}, \mathbf{F_B}) = \mathbf{0} \tag{117}$$

Forward Sensitivity Formulation The structural model disciplinary tangent (forward sensitivity) linearization can be constructed by considering a structural objective $L(\mathbf{Q}(D), D)$ and a design variable D. The sensitivity of the objective with respect to the design variable can be written as:

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial \mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial D} \tag{118}$$

This requires solving for the sensitivity of the structural model state ($\mathbf{Q}$), which can be obtained by differentiating equation (117) with respect to the design variable D and rearranging as:

$$\left[\frac{\partial \mathbf{J}}{\partial \mathbf{Q}}\right]\frac{\partial \mathbf{Q}}{\partial D} = -\frac{\partial \mathbf{J}}{\partial \mathbf{F_B}}\frac{\partial \mathbf{F_B}}{\partial D} - \frac{\partial \mathbf{J}}{\partial D} \tag{119}$$

Thus the objective sensitivity is obtained as:

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} - \frac{\partial L}{\partial \mathbf{Q}}\left[\frac{\partial \mathbf{J}}{\partial \mathbf{Q}}\right]^{-1}\left[\frac{\partial \mathbf{J}}{\partial \mathbf{F_B}}\frac{\partial \mathbf{F_B}}{\partial D} + \frac{\partial \mathbf{J}}{\partial D}\right] \tag{120}$$

which can be computed by solving equations (119) and substituting the structural sensitivities into equation (118)

Adjoint Formulation Alternatively, we may consider a structural adjoint variable defined as:

$$\left[\frac{\partial \mathbf{J}}{\partial \mathbf{Q}}\right]^T \mathbf{\Lambda_Q} = -\frac{\partial L}{\partial \mathbf{Q}}^T \tag{121}$$

and construct the objective sensitivity after the solution of this equation as:

$$\frac{dL}{dD} = \frac{\partial L}{\partial D} + \mathbf{\Lambda}_Q\left[\frac{\partial \mathbf{J}}{\partial \mathbf{F_B}}\frac{\partial \mathbf{F_B}}{\partial D} + \frac{\partial \mathbf{J}}{\partial D}\right] \tag{122}$$

In the forward and adjoint cases, the last term on the right hand side of equation (122) or (120) is non zero for structural design parameters such as element stiffnesses, in which case the applied force does not change with the design parameter, making the first term on the right hand size zero. In the coupled aeroelastic case, using aerodynamic shape parameters that primarily affect the airloads on the structure, the first term on the right-hand size is non-zero while the second term vanishes in the absence of structural design parameters.

4.4.2 Fluid-structure interface (FSI)

In addition to the solution of the aerodynamic problem and the structural dynamics problem, the solution of the fully coupled time-dependent aeroelastic problem requires the exchange of aerodynamic loads from the computational fluid dynamic (CFD) solver to the structural model, which in turn returns surface displacements to the fluid flow solver. The governing equations for the FSI can be written in residual form as:

$$\mathbf{S}(\mathbf{F_B}, \mathbf{Q}, \mathbf{F}(\mathbf{x}, \mathbf{u})) = \mathbf{F_B} - [T(\mathbf{Q})]\mathbf{F}(\mathbf{x}, \mathbf{u}) = 0 \tag{123}$$

$$\mathbf{S}'(\mathbf{x_s}, \mathbf{Q}) = \mathbf{x_s} - [T(\mathbf{Q})]^T\mathbf{Q} = 0 \tag{124}$$

respectively for the forces transferred to the structural solver and displacements returned to the flow solver. In these equations, $[T]$ represents the transfer matrix which projects point-wise CFD surface forces $\mathbf{F}(\mathbf{x}, \mathbf{u})$ onto the individual beam elements resulting in the beam forces $\mathbf{F_B}$. The transpose of this matrix is used to obtain the CFD surface displacements $\mathbf{x_s}$ from the beam degrees of freedom $\mathbf{Q}$. Also note that $[T]$ is a function of $\mathbf{Q}$ since the transfer patterns change with the beam deflection.

4.4.3 General solution procedure

The aeroelastic problem consists of multiple coupled sets of equations namely, the mesh deformation equations, the flow equations (CFD), the structural dynamic equations, and

the fluid-structure interface transfer equations. The system of equations to be solved at each time step can be written as:

$$\mathbf{G}(\mathbf{x}, \mathbf{x_s}(\mathbf{Q}), D) = 0 \tag{125}$$
$$\mathbf{R}(\mathbf{u}, \mathbf{x}) = 0 \tag{126}$$
$$\mathbf{S}(\mathbf{F_B}, \mathbf{Q}, \mathbf{F}(\mathbf{x}, \mathbf{u})) = 0 \tag{127}$$
$$\mathbf{J}(\mathbf{Q}, \mathbf{F_B}, D) = 0 \tag{128}$$
$$\mathbf{S'}(\mathbf{x_s}, \mathbf{Q}) = 0 \tag{129}$$

where $\mathbf{S}$ and $\mathbf{S'}$ represent the residuals of the FSI equations, and $\mathbf{J}$ represents the residual of the structural dynamics problem. Note that the mesh motion residual now depends also on any surface deflections $\mathbf{x_s}(\mathbf{Q})$ introduced by the structural model. We also consider two types of design variables, shape variables that affect directly the mesh configuration and appear in the first equation, and structural design variables such as element stiffnesses and structural geometry parameters that appear directly in the structural equations. In practice, these two classes of design parameters may overlap as external shape changes may also result in internal structural geometry changes.

For the analysis problem, the design parameters are held fixed and are thus inconsequential to the solution process. Within each physical time step, solution of the fully coupled fluid structure problem consists of performing multiple coupling iterations on each discipline using the latest available values from the other disciplines. Thus the coupled iteration strategy proceeds as:

$$\left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right] \Delta \mathbf{x}^c = -\mathbf{G}(\mathbf{x^{c-1}}, \mathbf{x_s^{c-1}}) \tag{130}$$
$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right] \Delta \mathbf{u}^c = -\mathbf{R}(\mathbf{u^{c-1}}, \mathbf{x^c}) \tag{131}$$

for the mesh and flow equations, where the superscript c denotes the coupling iteration index, and the variables are updated as $\mathbf{x^c} = \mathbf{x^{c-1}} + \Delta \mathbf{x^c}$ and $\mathbf{u^c} = \mathbf{u^{c-1}} + \Delta \mathbf{u^c}$. This is followed by the solution of the FSI and structural model as:

$$\mathbf{F_B^c} = [\mathbf{T}(\mathbf{Q^c})]\, \mathbf{F}(\mathbf{x^c}, \mathbf{u^c}) \tag{132}$$
$$\left[\frac{\partial \mathbf{J}}{\partial \mathbf{Q}}\right] \Delta \mathbf{Q}^c = -\mathbf{J}(\mathbf{Q^c}, \mathbf{F_B^c}) \tag{133}$$
$$\mathbf{x_s} = [\mathbf{T}(\mathbf{Q^c})]^T\, \mathbf{Q}^c \tag{134}$$

In this implementation, subiterations are performed to converge the first two equations simultaneously for the final values of $\mathbf{Q^c}$, while the third equation corresponds to an explicit evaluation for $\mathbf{x_s}$ given the $\mathbf{Q^c}$ values.

At the first coupling iteration, $\mathbf{x_s^c} = \mathbf{0}$ and solution of the mesh deformation equation is trivial, although non zero values of $\mathbf{x_s}$ are produced at subsequent coupling iterations as the beam deflects in response to the aerodynamic loads. From a disciplinary point of view, the aerodynamic solver produces updated values of $\mathbf{u}$ and $\mathbf{x}$, which are used to compute $\mathbf{F}(\mathbf{x}, \mathbf{u})$ pointwise CFD surface forces. These surface forces are input to the FSI/structural model which returns surface displacements $\mathbf{x_s}$. These new surface displacements are then

fed back into the mesh deformation equations and the entire procedure is repeated until convergence is obtained for the full coupled aero-structural problem at the given time step. This approach corresponds to a disciplinary Gauss-Seidel strategy, which is suitable for a wide range of aeroelastic problems. However, for highly flexible structures, more tightly coupled Newton-Krylov methods may be required to guarantee coupled disciplinary convergence, although these will not be considered in the current context for simplicity.

4.4.4 Sensitivity Analysis for Coupled Aeroelastic Problem

In the formulation of the sensitivity analysis for the coupled aeroelastic problem, it is desirable to mimic as closely as possible the solution strategies and data structures employed for the analysis problem. Thus, analogous disciplinary solvers can be reused for each disciplinary sensitivity problem, and the analysis coupling strategy can be extended to the sensitivity analysis formulation. Furthermore, the data transferred between disciplinary solvers should consist of vectors of the same dimension for the analysis, tangent and adjoint formulations. Following the approach described previously for multidisciplianry problems, the sensitivity of a purely aerodynamic objective L can be written as:

$$\frac{dL}{dD} = \begin{bmatrix} \dfrac{\partial L}{\partial \mathbf{x}} & \dfrac{\partial L}{\partial \mathbf{u}} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathbf{x}}{\partial D} \\ \dfrac{\partial \mathbf{u}}{\partial D} \end{bmatrix} \tag{135}$$

since the aerodynamic objective depends only directly on the outer mold line ($\mathbf{x}$) and the flow field ($\mathbf{u}$). However, the individual disciplinary sensitivities are given as the solution of the coupled system:

$$\begin{bmatrix} \dfrac{\partial \mathbf{G}}{\partial \mathbf{x}} & 0 & 0 & 0 & 0 & \dfrac{\partial \mathbf{G}}{\partial \mathbf{x_s}} \\ \dfrac{\partial \mathbf{R}}{\partial \mathbf{x}} & \dfrac{\partial \mathbf{R}}{\partial \mathbf{u}} & 0 & 0 & 0 & 0 \\ -\dfrac{\partial \mathbf{F}}{\partial \mathbf{x}} & -\dfrac{\partial \mathbf{F}}{\partial \mathbf{u}} & I & 0 & 0 & 0 \\ 0 & 0 & \dfrac{\partial \mathbf{S}}{\partial \mathbf{F}} & \dfrac{\partial \mathbf{S}}{\partial \mathbf{F_B}} & \dfrac{\partial \mathbf{S}}{\partial \mathbf{Q}} & 0 \\ 0 & 0 & 0 & \dfrac{\partial \mathbf{J}}{\partial \mathbf{F_B}} & \dfrac{\partial \mathbf{J}}{\partial \mathbf{Q}} & 0 \\ 0 & 0 & 0 & 0 & \dfrac{\partial \mathbf{S}'}{\partial \mathbf{Q}} & \dfrac{\partial \mathbf{S}'}{\partial \mathbf{x_s}} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathbf{x}}{\partial D} \\ \dfrac{\partial \mathbf{u}}{\partial D} \\ \dfrac{\partial \mathbf{F}}{\partial D} \\ \dfrac{\partial \mathbf{F_B}}{\partial D} \\ \dfrac{\partial \mathbf{Q}}{\partial D} \\ \dfrac{\partial \mathbf{x_s}}{\partial D} \end{bmatrix} = \begin{bmatrix} -\dfrac{\partial \mathbf{G}}{\partial D} \\ 0 \\ 0 \\ 0 \\ -\dfrac{\partial \mathbf{J}}{\partial D} \\ 0 \end{bmatrix}$$

The first and second equations correspond to equations for the mesh and flow variable sensitivities, as previously described for the aerodynamic solver, and the third equation corresponds to the construction of the CFD surface force sensitivities given these two

previous sensitivities. The fourth equation denotes the sensitivity of the FSI transfer from the fluid to the structural solver, while the fifth equation corresponds to the sensitivity of the structural solver. Finally, the last equation corresponds to the sensitivity of the FSI transfer from the structural solver back to the flow solver. This coupled system of sensitivities can be solved analogously to the coupled analysis problem as:

$$\left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right] \frac{\partial \mathbf{x}}{\partial \mathbf{D}}^{c} = -\frac{\partial \mathbf{G}}{\partial \mathbf{x_s}} \frac{\partial \mathbf{x}}{\partial \mathbf{D}}^{\mathbf{c-1}} - \frac{\partial \mathbf{G}}{\partial \mathbf{D}} \tag{136}$$

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right] \frac{\partial \mathbf{u}}{\partial \mathbf{D}}^{\mathbf{c}} = -\frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{D}}^{\mathbf{c}} \tag{137}$$

followed by the explicit evaluation of the surface force sensitivities as:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{D}}^{\mathbf{c}} = \frac{\partial \mathbf{F}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{D}}^{\mathbf{c}} + \frac{\partial \mathbf{F}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{D}}^{\mathbf{c}} \tag{138}$$

These sensitivity evaluations are followed by the solution of the remaining components of the system as:

$$\begin{bmatrix} \frac{\partial \mathbf{S}}{\partial \mathbf{F_B}} & \frac{\partial \mathbf{S}}{\partial \mathbf{Q}} \\ \frac{\partial \mathbf{J}}{\partial \mathbf{F_B}} & \frac{\partial \mathbf{J}}{\partial \mathbf{Q}} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathbf{F_B}}{\partial D}^{c} \\ \frac{\partial \mathbf{Q}}{\partial D}^{c} \end{bmatrix} = \begin{bmatrix} -\frac{\partial \mathbf{S}}{\partial \mathbf{F}} \frac{\partial \mathbf{F}}{\partial D}^{c} \\ -\frac{\partial \mathbf{J}}{\partial D} \end{bmatrix} \tag{139}$$

These two equations are solved simultaneously to obtain the structural sensitivities $\frac{\partial \mathbf{Q}}{\partial D}$ which are then used to evaluate the surface mesh sensitivities explicitly as:

$$\frac{\partial \mathbf{x_s}}{\partial D}^{c} = -\frac{\partial \mathbf{S'}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial D}^{c} \tag{140}$$

where the fact that $\frac{\partial \mathbf{S'}}{\partial \mathbf{x_s}} = [I]$ (identity matrix) has been used. These new surface mesh sensitivities are then fed back into the first equation in the system initiating the next coupling iteration. As can be seen, each disciplinary solution procedure requires the inversion of the same Jacobian matrix as the corresponding analysis problem, which is done using the same iterative solver. Furthermore, the fluid-structure coupling requires the transfer of the force sensitivities $\frac{\partial \mathbf{F}}{\partial D}$ from the flow to the structural solver, and the surface mesh sensitivities $\frac{\partial \mathbf{x_s}}{\partial D}$ from the structural solver back to the fluid solver, which are of the same dimension as the force and surface displacements transferred in the analysis problem, respectively.

The corresponding adjoint problem can be obtained by transposing the coupled Jacobian matrix and defining adjoint variables as solutions to the following coupled system (following the derivation in section 4.2):

$$\begin{bmatrix} \frac{\partial \mathbf{G}}{\partial \mathbf{x}}^T & \frac{\partial \mathbf{R}}{\partial \mathbf{x}}^T & -\frac{\partial \mathbf{F}}{\partial \mathbf{x}}^T & 0 & 0 & 0 \\ 0 & \frac{\partial \mathbf{R}}{\partial \mathbf{u}}^T & -\frac{\partial \mathbf{F}}{\partial \mathbf{u}}^T & 0 & 0 & 0 \\ 0 & 0 & I & \frac{\partial \mathbf{S}}{\partial \mathbf{F}}^T & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial \mathbf{S}}{\partial \mathbf{F_B}}^T & \frac{\partial \mathbf{J}}{\partial \mathbf{F_B}}^T & 0 \\ 0 & 0 & 0 & \frac{\partial \mathbf{S}}{\partial \mathbf{Q}}^T & \frac{\partial \mathbf{J}}{\partial \mathbf{Q}}^T & \frac{\partial \mathbf{S'}}{\partial \mathbf{Q}}^T \\ \frac{\partial \mathbf{G}}{\partial \mathbf{x_s}}^T & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{S'}}{\partial \mathbf{x_s}}^T \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_x} \\ \mathbf{\Lambda_u} \\ \mathbf{\Lambda_F} \\ \mathbf{\Lambda_{F_B}} \\ \mathbf{\Lambda_Q} \\ \mathbf{\Lambda_{x_s}} \end{bmatrix} = - \begin{bmatrix} \frac{\partial L}{\partial \mathbf{x}}^T \\ \frac{\partial L}{\partial \mathbf{u}}^T \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This system can be solved starting with the last equation and proceeding to the first equation as:

$$\mathbf{\Lambda_{x_s}}^c = -\frac{\partial \mathbf{G}}{\partial \mathbf{x_s}}^T \mathbf{\Lambda_x}^{c-1} \tag{141}$$

(again using the fact that $\frac{\partial \mathbf{S'}}{\partial \mathbf{x_s}}$ corresponds to the identity matrix), followed by the solution of the structural adjoints

$$\begin{bmatrix} \frac{\partial \mathbf{S}}{\partial \mathbf{F_B}}^T & \frac{\partial \mathbf{J}}{\partial \mathbf{F_B}}^T \\ \frac{\partial \mathbf{S}}{\partial \mathbf{Q}}^T & \frac{\partial \mathbf{J}}{\partial \mathbf{Q}}^T \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_{F_B}}^c \\ \mathbf{\Lambda_Q}^c \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{\partial \mathbf{S'}}{\partial \mathbf{Q}}^T \mathbf{\Lambda_{x_s}}^c \end{bmatrix} \tag{142}$$

followed by the explicit construction of the pointwise CFD surface force adjoint:

$$\mathbf{\Lambda_F}^c = -\frac{\partial \mathbf{S}}{\partial \mathbf{F}}^T \mathbf{\Lambda_{F_B}}^c \tag{143}$$

and ending with the solution of the mesh and flow adjoints as:

$$\begin{bmatrix} \frac{\partial \mathbf{G}}{\partial \mathbf{x}^T} & \frac{\partial \mathbf{R}}{\partial \mathbf{x}^T} \\ 0 & \frac{\partial \mathbf{R}}{\partial \mathbf{u}^T} \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_x}^c \\ \mathbf{\Lambda_u}^c \end{bmatrix} = \begin{bmatrix} \frac{\partial L}{\partial \mathbf{x}}^T + \frac{\partial \mathbf{F}}{\partial \mathbf{x}}^T \mathbf{\Lambda_F}^c \\ \frac{\partial L}{\partial \mathbf{u}}^T + \frac{\partial \mathbf{F}}{\partial \mathbf{u}}^T \mathbf{\Lambda_F}^c \end{bmatrix} \tag{144}$$

Once again, the solution of the various disciplinary adjoints requires the inversion of the corresponding disciplinary Jacobians (transposed in this case) which can be accomplished using the same iterative solvers as for the analysis and forward sensitivity problems.

Additionally, the input to the structural adjoint problem consists of the variable $\mathbf{\Lambda_{x_s}}$, which is the same dimension as the surface displacements output from the structural analysis solver, while the output of the structural adjoint solver consists of the variable $\mathbf{\Lambda_F}$ which is of the same dimension as the force inputs to the structural solver in the analysis problem. The interdisciplinary coupling between the aerodynamics and structural dynamics disciplines is illustrated in Figure 7, where the flow of information in the case of the adjoint solution is seen to be the reverse of the information flow in the corresponding analysis and forward sensitivity cases.

Once the coupled adjoint problem has been solved, the final objective sensitivity vector can be accumulated as:

$$\frac{dL}{dD}^T = \begin{bmatrix} \frac{\partial \mathbf{G}}{\partial D}^T & 0 & 0 & 0 & \frac{\partial \mathbf{J}}{\partial D}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda_x} \\ \mathbf{\Lambda_u} \\ \mathbf{\Lambda_F} \\ \mathbf{\Lambda_{F_B}} \\ \mathbf{\Lambda_Q} \\ \mathbf{\Lambda_{x_s}} \end{bmatrix} \tag{145}$$

We note that, for this particular problem, only $\mathbf{\Lambda_x}$ and $\mathbf{\Lambda_Q}$ are required in the final inner product. Additionally, if only aerodynamic design variables are present, then only $\mathbf{\Lambda_x}$ contributes to the final sensitivities. However, all the adjoint variables are coupled and thus influence the final sensitivities indirectly.

4.4.5 Verification of Coupled Aeroelastic Sensitivity

One of the advantages of discrete versus continuous adjoint implementations is that the sensitivities produced by the discrete adjoint technique can be verified exactly for correctness. This is due to the fact that the sensitivities produced by the discrete adjoint approach correspond to the exact sensitivities of the simulation code, and in principle should be reproducible by finite differencing the simulation code, i.e. by perturbing each input parameter and observing the change in the simulation output or objective as a function of the input perturbation. However, it is well known that the finite differencing procedure itself is prone to error which varies as a function of the perturbation size. In order to minimize non-linear error, small perturbation sizes are required, although these can lead to large round-off error in the finite-difference procedure. While agreement to several significant figures often can be achieved between finite-differencing and adjoint

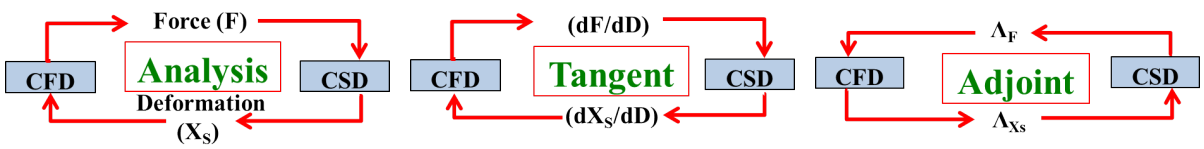


Figure 7: Illustration of aero-structural coupling for: analysis problem (left), forward sensitivity problem (center), adjoint problem (right)

derived sensitivities, experience has shown that coding errors can persist undetected at this level only to be amplified in later simulations where linearizations about different states occur. This situation is exacerbated as the simulation complexity increases, since individual components or disciplines may have conflicting accuracy requirements making the selection of a single overall perturbation value difficult or impossible.

Fortunately, there are several ways to obtain exact discrete gradients for verification purposes. For example, in the complex step method [10], all variables and functions are redefined to be complex types and the input parameters are perturbed only in their complex component. Noting that the Taylor series of a complex function with complex perturbation can be written as:

$$f(x + ih) = f(x) + i\epsilon f'(x) + \cdots \tag{146}$$

the derivative $f'(x)$ can be easily determined as:

$$f'(x) = \frac{Im\left[f\left(x + i\epsilon\right)\right]}{\epsilon} \tag{147}$$

Since the derivative value appears directly in the imaginary part of the perturbed function, the subtractive cancellation error of the finite difference procedure is avoided and arbitrarily small perturbation values ϵ may be used.

Closely related approaches involve the use of dual numbers with operator overloading [2] and automatic differentiation [1, 6]. In all cases, these techniques rely on source code transformation which can be automated to prevent coding errors and produce exact derivatives of any computed quantity with respect to the selected input parameters. In fact, any of these approaches represents a viable technique for computing forward sensitivities in the absence of an exact coded tangent implementation, and are often preferred due to their ease of implementation. On the other hand, although automatic (reverse) differentiation for adjoint methods are available, these tend to come at substantial increased cost in terms of memory use and computational expense, which can increase rapidly with simulation complexity.

In the current implementation, we use the complex step method to verify the sensitivities produced by the tangent implementation. For complex multidisciplinary simulations, verification can be done at the unit testing level, built up to the component or disciplinary level, and ultimately to the entire time-dependent multidisciplinary level. This is achieved by maintaining a complex version of the complete simulation source code, which is invoked through automated source code transformation at compile time to enable the verification tests.

Once the tangent sensitivities have been verified, the corresponding adjoint sensitivities are verified by comparing these with the tangent and complex values. Using the duality relation, these can be compared also at the unit testing level, and built up to the component or disciplinary level and finally to the entire time-dependent multidisciplinary level [25, 15]. Because these verification tests are exact, agreement to machine precision is expected between the complex step, tangent and adjoint sensitivities at all stages of the simulation process.

5 Time-Dependent Aeroelastic Optimization of a Helicopter Rotor

5.1 Component Solvers

The simulation of a flexible (aeroelastic) helicopter rotor in forward flight is chosen as a test case to demonstrate the implementation and use of the time-dependent aeroelastic adjoint formulation described in the previous section. In addition to being time-dependent, the forward flight rotor test case includes prescribed solid body motion (rotation of the rotor), prescribed relative motion (individual blade cyclic pitching), and flexible blade deflection computed as part of the structural analysis solution. Furthermore, the design process makes use of both blade shape parameters and rotor control parameters (used to control the prescribed blade pitching motion).

The test case consists of a four bladed HART-II (Higher Harmonic Control Aeroacoustic Rotor Test) rotor [37] in a forward flight condition. The flight conditions include a freestream Mach number of $M_\infty = 0.095$, a blade tip Mach number of $M_{tip} = 0.638$ and a shaft tilt angle towards freestream of $\alpha_{shaft} = 5.4°$. The diameter of the rotor is 4 meters, and the corresponding rotor rotational speed is $\Omega = 1041$ RPM. In addition to the rotational motion, the individual blades undergo a prescribed pitching motion as a function of azimuthal location given as: $\theta = \theta_0 + \theta_{1s}sin(\psi) + \theta_{1c}cos(\psi)$, with $\theta_0 = 5.0°, \theta_{1s} = -1.1°, \theta_{1c} = 2.0°$. where θ is the prescribed blade pitch angle and ψ corresponds to the azimuthal location of the blade about the hub vertical rotation axis. The rotor is impulsively started from rest, in an initially quiescent flow field, and rotated with the mesh as a solid body for a fixed number of revolutions. However, at each time step the interior mesh must be deformed in response to the prescribed blade pitching motion and flexible structural response.

Simulation of this test case requires four disciplinary modules: a fluid flow (CFD) solver, a structural model, a fluid structure interface, and a mesh deformation solver. The NSU3D unstructured mesh Reynolds-averaged Navier-Stokes flow solver [17, 18, 19] is used for the CFD component in this case. NSU3D operates on mixed element unstructured meshes and utilizes a line-implicit agglomeration multigrid solver for fast convergence at each implicit time step. The flow equations are solved in ALE form and the Spalart-Allmaras turbulence model [33] is used in all cases. NSU3D has also been well validated for rotorcraft aerodynamic problems through its use as a component in the HELIOS rotorcraft simulation software suite developed by the US Department of Defense [31]. NSU3D also contains discrete unsteady aerodynamic tangent and adjoint capabilities which have been verified exactly and demonstrated in previous work [15, 16, 11, 24].

The blade structure is approximated by a non-linear bend-twist beam model, which is suitable for slender fixed and rotary wing aircraft structures. As depicted in Figure 6, the finite-element beam model includes 15 degrees of freedom for each element to accommodate bend-wise, lag-wise, axial and torsional displacements. The discrete equations of motion resulting from the finite-element beam model discretization, as described in equations (117), are solved in parallel by direct inversion with prefactorization. The beam model has been validated [22] for the standard HART-II rotor case [37] by comparing its natural frequency predictions with the predictions from other reliable computational structural dynamics (CSD) models [32]. Additionally, the discrete structural tangent and

adjoint models derived in equations (119) and (121) have been implemented and fully verified for the beam structural model.

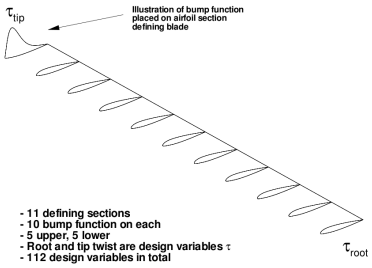
The fluid structure interface (FSI) is implemented by treating the surface CFD mesh points for each blade as a cloud of points with no connectivity information and projecting each point to the closest element of the beam model based on the minimum normal distance in order to build the transfer matrix $[T]$ defined in equations (123) and (124).

The mesh deformation solver is based on a linear elastic analogy where the mesh is treated as a linear elastic solid which responds to displacements imposed at the blade surface mesh points [36]. The approach uses a variable modulus of elasticity which can be prescribed either as inversely proportional to the mesh cell volumes or to the distance from the closest wall, thus minimizing cell distortion for small elements and in near wall regions. The resulting mesh motion equations are solved by reusing the line-implicit agglomeration multigrid algorithm employed by the flow solver.

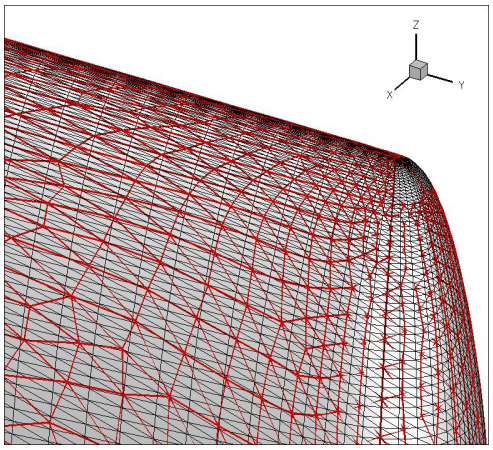
5.2 Geometry Parameterization

In order to obtain sensitivities with respect to a set of shape parameters that are well suited for design optimization purposes, a baseline blade is constructed by stacking 11 airfoil sections along the span. Each airfoil contains 10 Hicks-Henne bump functions, 5 on the upper surface, and 5 on the lower surface, that can be used to modify the airfoil shape. Additionally, the twist values of the blade at the root and tip airfoil sections are also used as design variables resulting in a total of 112 design variables. Figure 8.1 provides an illustration of the baseline blade design setup. A high density structured mesh is generated about this blade geometry, which is then rotated and translated to match each individual blade in the CFD mesh, as shown in Figure 8.2. Interpolation patterns between each unstructured mesh surface point and the baseline structured mesh are determined in a preprocessing phase. These interpolation patterns are then used to interpolate shape changes from the baseline blade to all four blades in the CFD mesh (as determined by changes in the design variables) and to transfer sensitivities from the surface CFD mesh points to the design variables using the chain rule of differentiation.

In addition to the aforementioned blade shape design variables, the cyclic pitching control parameters are also included as design parameters. This results in three additional parameters, $\theta_{\mathbf{0}}$ the rotor collective pitch angle, and $\theta_{\mathbf{c_i}}$ and $\theta_{\mathbf{s_i}}$ the rotor cyclic pitch angles. These parameters are used to define the prescribed blade pitching motion as described previously, and can be varied throughout the design process in order to increase the design space. This results in a total of 115 design variables. These sensitivities, in addition to the above mentioned geometric sensitivities, are transferred onto the CFD surface mesh points from the blade defining surface through the interpolation method described earlier. We note that in this test case we only consider aerodynamic shape design variables, and do not include any structural design variables. Furthermore, we assume that shape changes to the outer mold line of the blades do not affect the structural model. Thus, the term $\frac{\partial \mathbf{J}}{\partial D}$ in equation (145) vanishes. Although the design variables do not directly influence the structural model, the blade structural response affects the aerodynamic response of the rotor and thus the design outcome.



8.1: Blade design parameters



8.2: Baseline structured blade mesh overlap with CFD mesh

Figure 8: Illustration of (left) baseline blade with design parameters and (right) overlap in tip region between baseline blade surface definition and CFD surface unstructured mesh.

5.3 Analysis Problem

Thr rotor analysis problem is solved both for a rigid blade model (using no structural model), as well as for a flexible blade model (using the beam structural model). For the latter, the flow is solved in tight coupling mode, where the fully coupled fluid-structure equations are converged at each time step using the disciplinary Gauss-Seidel approach described earlier.

The baseline simulation makes use of a mixed element mesh made up of prisms, pyramids and tetrahedra consisting of approximately 2.32 million grid points and is shown in Figure 9 along with an illustration of the prescribed rotational and cyclic pitching motion of the rotor blades. The simulations are run for 5 rotor revolutions using a 2 degree time-step, i.e. for 900 time-steps starting from freestream initialization.

For the rigid blade simulation, the time-dependent mesh motion is determined by first pitching the blade about the blade axis followed by solving the mesh deformation equations and then rotating the entire mesh as a solid body at each time step.

The coupled CFD/CSD simulation is run in a similar manner. However, the flow

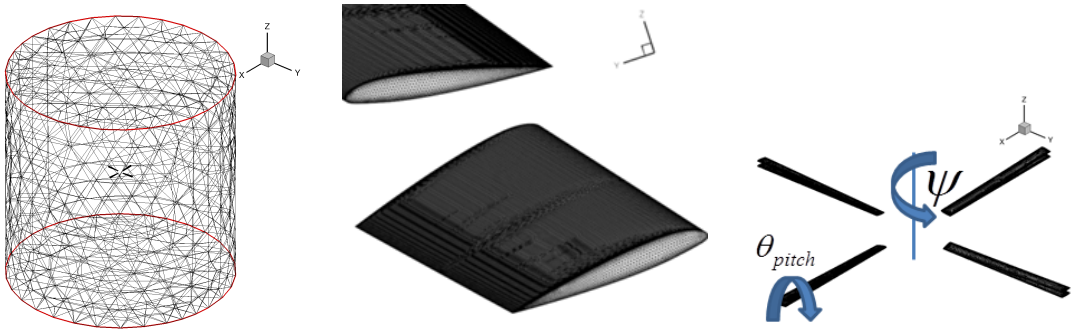


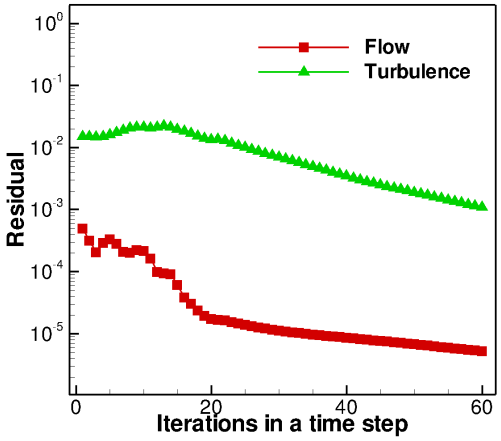
Figure 9: Computational mesh for baseline HART-II rotor in forward flight test case and illustration of prescribed pitch and rotation blade motion.

solution (CFD) is coupled with the beam solver (CSD) at every time step by appropriately exchanging, a) airloads information from the flow domain to the beam and b) blade deformation information from the beam to the flow domain, through the fluid-structure interface (FSI). In this coupled simulation, the mesh is moved according to the deformations dictated by the new flexed blade coordinates determined from the structural beam code after the combined kinematics of pitch actuation followed by solid body rotation of the entire mesh have been performed. Thus, the flow now sees not only the pitched and rigidly rotated mesh (observed in rigid blade simulation in forward flight), but also the deformed mesh around the blades due to both pitching and structural deformations. This coupled fluid-structure interaction problem needs to be iterated until satisfactory convergence is achieved on flow, structure and mesh deformation problems within each time step.

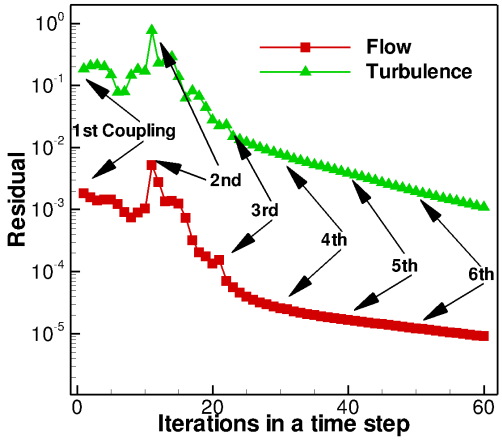
The simulations were performed on the Yellowstone supercomputer at the NCAR-Wyoming Supercomputing Center (NWSC), with the analysis problem running in parallel on 1024 cores. For the flexible rotor case, each time step used 6 coupling iterations, and each coupling iteration used 10 non-linear flow iterations with each non-linear iteration consisting of a three-level line-implicit multigrid cycle. The rigid rotor case employed the same overall number of multigrid cycles per time step (60) with no outer coupling iterations. The typical simulation at this level of resolution requires approximately 40 minutes of wall clock time per rotor revolution for both flexible and rigid rotor cases.

Figures 10 and 11 summarize the overall convergence of the rigid and aero-elastic coupling analysis formulations. Figure 10.1 shows the typical flow and turbulence residual convergence within a single time step for the rigid rotor case (no structural model), while Figure 10.2 depicts convergence of the flow and turbulence residuals at the same time step for the coupled aeroelastic case. In this case, the jumps in residual values at the start of new coupling iterations are clearly visible, although these jumps become smaller as the coupling procedure converges. The overall residual histories closely follow those of the rigid rotor case after the first few coupling iterations. Figure 11.1 depicts the convergence of the mesh deformation residual for the same time step, also showing jumps in the residual at the start of each new coupling iteration. Solution of the mesh deformation equations terminates when the residuals reach a prescribed tolerance of 10^{-6}, and hence the variable number of iterations per coupling cycle. Most notable is the fact that the initial mesh deformation residual decreases at each new coupling iteration, providing a measure of the convergence of the entire coupling procedure. The corresponding beam structural model residual drop is observed to be of 15 orders of magnitude, as shown in Figure 11.2. This figure illustrates the convergence of the coupled beam/FSI residual (i.e. equations (127) and (128)), showing rapid convergence to machine zero in a small number of iterations within a single CFD/CSD coupling iteration.

The effect of the CFD/CSD aeroelastic coupling is clearly demonstrated in Figure 12.1, which compares the deformed blade shape and its corresponding C_p surface contours from the coupled simulation with that from the rigid blade simulation. From the figure it is noted that all four blades show different deformation characteristics due to the corresponding different aerodynamic environment they experience in a forward flight. The blade attains the largest flap displacement at azimuth, $\psi = 180°$ and the smallest flap displacement at azimuth, $\psi = 0°$. For both the flexible and rigid blades, the pressure contours demonstrate that the advancing side blades experience more compressibility ef-

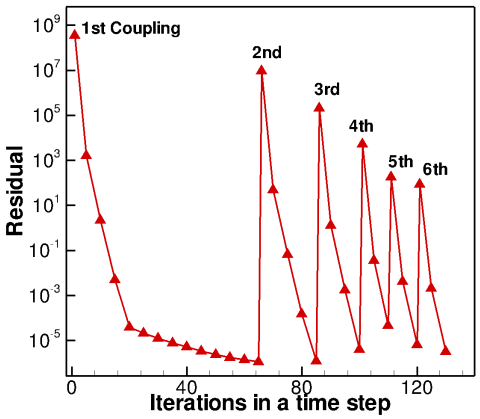


10.1: Rigid Blade

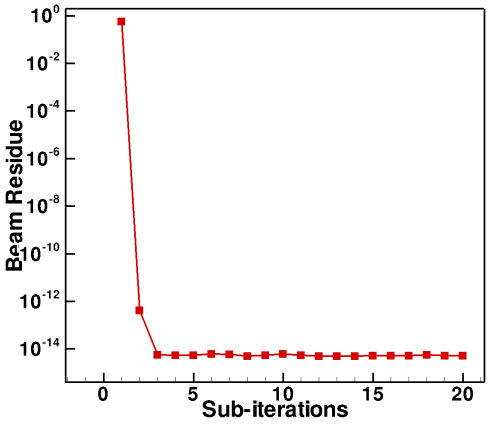


10.2: Flexible Blade

Figure 10: Flow and turbulence residual convergence at a given time step for rigid and flexible blade analysis

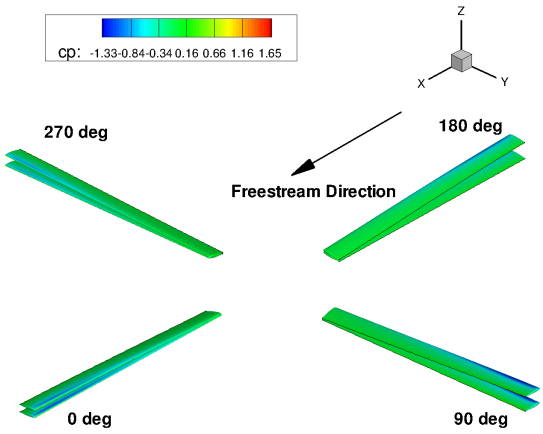


11.1: Mesh deformation convergence

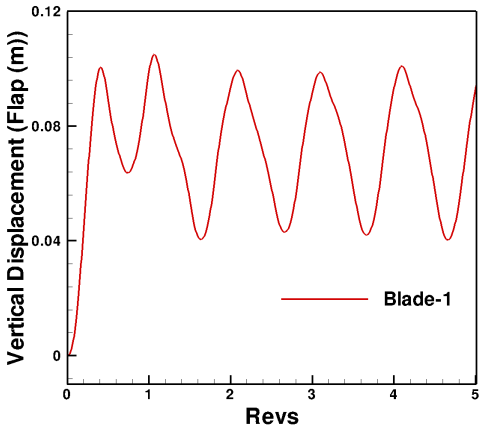


11.2: Beam and FSI convergence

Figure 11: Residual convergence of (a) mesh deformation equations for coupled analysis and (b) beam (overall FSI) within a single coupling iteration



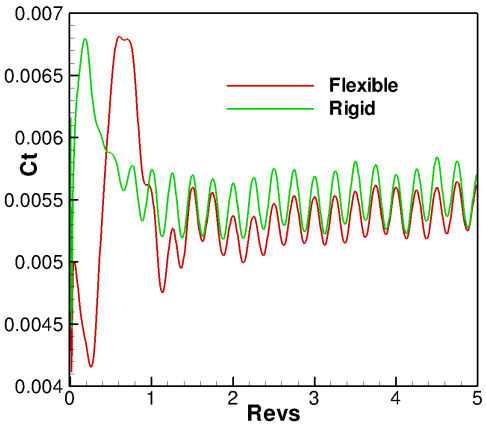
12.1: Blade Deformation (C_p contours)



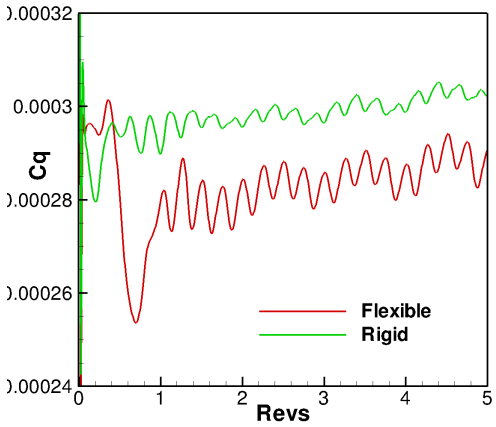
12.2: Blade-1 tip vs time

Figure 12: Illustration of HART-II flexible blade deformation analysis results

fects (larger pressure gradients near the rotor tips) than the retreating side blades. The flexible blade tip vertical displacement time history shown in Figure 12.2 demonstrates a periodic flapping motion with a period of one rotor revolution ($1/rev$).



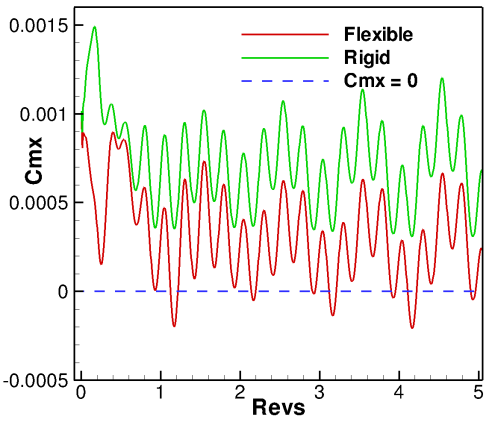
13.1: Thrust coefficient vs time



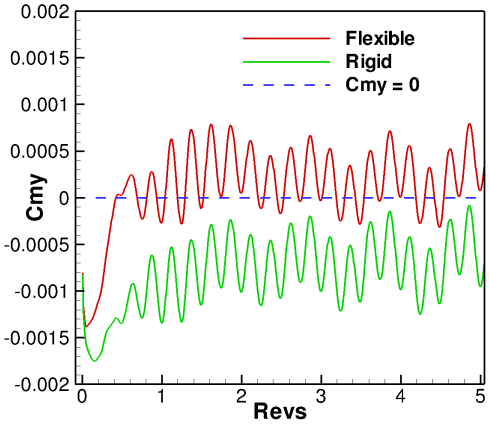
13.2: Torque coefficient vs time

Figure 13: Performance comparison of flexible blade with rigid blade

Figures 13 summarizes the performance comparison between flexible and rigid blades over 5 rotor revolutions. Besides showing consistent 4/rev harmonic content in the airloads (as expected for a 4-bladed rotor), the figures also show that the rigid blades over predict thrust and power compared to flexible blades. This suggests that performance predictions might be erroneous if the aeroelastic nature of rotor blade is not taken into account. Figures 14 further compare the rotor longitudinal (C_{M_Y}) and lateral (C_{M_X}) moments of the flexible rotor with the rigid one. It should be noted that neither the rigid nor the flexible rotor are in a moment trimmed state (zero mean moment) at the prescribed pitch parameters.



14.1: Lateral moment vs time



14.2: Longitudinal moment vs time

Figure 14: Rotor moments comparison of flexible blade with rigid blade

5.4 Unsteady Objective Formulation

A time-integrated objective function based on the time variation of the thrust (C_T), torque (C_Q) and moment coefficients (C_{M_X} and C_{M_Y}) are used for this test case. The goal of the optimization is to reduce the time-integrated torque coefficient while constraining the time-integrated thrust coefficient to the baseline rotor performance as well as constraining the moment coefficients along roll (C_{M_X}) and pitch (C_{M_Y}) axes of the rotor to a trimmed value, i.e. zero moment values. The objective function is based on the summation of the differences between a target and a computed objective value at each time level n . Mathematically the global objective function is defined as:

$$L^g = L_{Shape} + L_{Trim} \tag{148}$$

$$L_{Shape} = \omega_1 \frac{1}{T} \sum_{n=1}^{n=N} \Delta t \left[\delta C_Q^n\right]^2 \tag{149}$$

$$L_{Trim} = \omega_2 \frac{1}{T} \sum_{n=1}^{n=N} \Delta t \left[\delta C_T^n\right]^2 + \omega_3 \left[\delta \bar{C}_{M_X}\right]^2 + \omega_4 \left[\delta \bar{C}_{M_Y}\right]^2 \tag{150}$$

$$\delta C_Q^n = (C_Q^n - C_{Q_{target}}^n) \tag{151}$$

$$\delta C_T^n = (C_T^n - \bar{C}_{T target}) \tag{152}$$

$$\delta \bar{C}_{M_X} = \frac{1}{T} \sum_{n=1}^{n=N} \Delta t (C_{M_X}^n - C_{M_X target}^n) \tag{153}$$

$$\delta \bar{C}_{M_Y} = \frac{1}{T} \sum_{n=1}^{n=N} \Delta t (C_{M_Y}^n - C_{M_Y target}^n) \tag{154}$$

where the mean target thrust coefficient value is specified to the HART-II baseline mean trim value of $\bar{C}_{T target} = 0.0044$, and the target torque and moment values are set to zero. The weights ($\omega_i, i = [1, 2, 3, 4]$) are included to equalize the difference in orders-of-magnitude between the thrust, torque and moment coefficients. Use of pitch control

parameters ($[\theta_0, \theta_{c_i}, \theta_{s_i}]$) as design variables and use of moment penalty terms ensure that the optimized rotor shape with optimized control parameters tends towards a final trimmed state when the rotor design cycles converge. This approach is computationally more efficient than the use of a hard constraint formulation, since this latter approach requires the computation of multiple adjoint problems at each design cycle (one for each additional constraint), as opposed to the single adjoint required in the current formulation. However, the trim constraint may not be satisfied exactly in this approach.

5.5 Unsteady Adjoint Sensitivity Verification

The fully coupled CFD/CSD adjoint formulation was verified by comparing its sensitivities with those obtained from the tangent as well as the complex step method. The coupled adjoint formulation was verified for perturbations on one geometric design parameter, namely, collective pitch (θ_0), for both uncoupled (rigid blade) as well as coupled (flexible blade) simulations. A complex perturbation of size 1×10^{-100} is introduced on collective pitch at the beginning of the analysis run. The design sensitivities of the functional ($\frac{\partial L^g}{\partial D}$) obtained from complex, tangent and adjoint formulations are compared after every time instance for up to 180 time steps in Table 1, which corresponds to a full rotor revolution (using 2^o time steps). Both the rigid and the fully coupled aeroelastic problems are converged to machine zero at each time step in order to avoid contaminating the sensitivity values with algebraic errors due to incomplete convergence. As can be seen from the table, the sensitivities agree to at least 11 significant digits for both the rigid and flexible aeroelastic rotor cases, and the agreement between the adjoint, tangent and complex methods does not degrade significantly with the number of time steps.

5.6 Performance Optimization of Helicopter Rotor in Forward Flight

The optimization tool thus verified is applied to the flexible HART-II rotor using the same time-dependent test case as described in the analysis section. The simulation is run for two full rotor revolutions, starting impulsively from rest in quiescent flow. The objective consists of the time-integrated torque with thrust and moment constraints applied using the penalty formulation, as described previously. However, the objective to be minimized is only integrated over the last one revolution, in order to avoid the optimization process from focusing on start-up transients. The optimization problem is solved on the same 2.3 million point unstructured mesh shown in Figure 9 using a time step size of 2.0 degrees. Figures 15 shows the residual convergence for a typical unsteady adjoint solution at a representative time step in the backwards time-integration process. The figure shows the residual drops by 5 orders of magnitude over 6 coupling cycles.

The optimization procedure used is the L-BFGS-B bounded reduced Hessian algorithm [38]. Each request by the optimization driver for a function and gradient value results in a single forward time-integration of the analysis solver and a single backward integration in time of the adjoint solver. A bound of $\pm 5\%$ chord for each defining airfoil section is set on the Hicks-Henne bump functions, a bound of $\pm 1.0^\circ$ of twist is set on the

Table 1: Adjoint sensitivity verification for Forward Flight

n	Method	Uncoupled (Rigid)	Coupled (Aero-elastic)
1	Complex	8.135662924562**930**E-005	7.569817143673**123**E-005
	Tangent	8.135662924562**923**E-005	7.569817143673**061**E-005
	Adjoint	8.135662924562**884**E-005	7.569817143672**761**E-005
2	Complex	7.303561433847**516**E-005	6.040142774935**852**E-005
	Tangent	7.303561433847**498**E-005	6.040142774935**835**E-005
	Adjoint	7.303561433847**042**E-005	6.040142774935**570**E-005
3	Complex	1.57907571578**3277**E-005	-4.95990987078**6381**E-006
	Tangent	1.57907571578**3178**E-005	-4.95990987078**7765**E-006
	Adjoint	1.57907571578**0985**E-005	-4.95990987078**5228**E-006
4	Complex	-3.279789405**316752**E-005	-6.193896187**819846**E-005
	Tangent	-3.279789405**316760**E-005	-6.193896187**820087**E-005
	Adjoint	-3.279789405**324491**E-005	-6.193896187**820419**E-005
5	Complex	-7.4662642948**60007**E-005	-1.142069116982**308**E-004
	Tangent	-7.4662642948**59811**E-005	-1.142069116982**308**E-004
	Adjoint	-7.4662642948**78836**E-005	-1.142069116982**432**E-004
180	Complex	-2.606225487**510254**E-003	-5.176189427**439016**E-003
	Tangent	-2.606225487**510356**E-003	-5.176189427**439005**E-003
	Adjoint	-2.606225487**528119**E-003	-5.176189427**434507**E-003

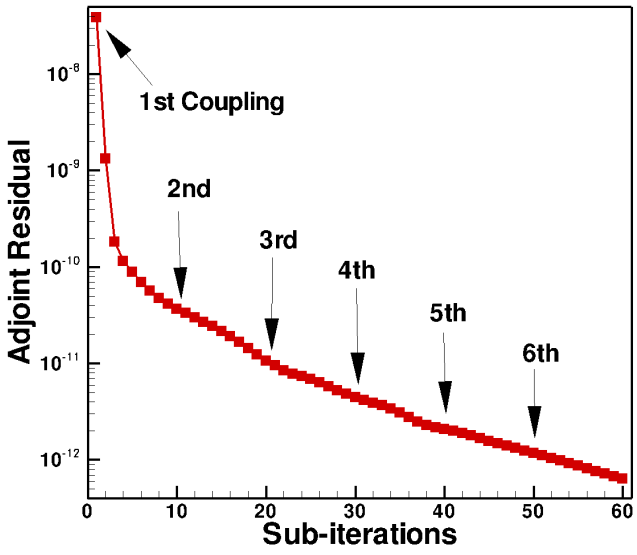


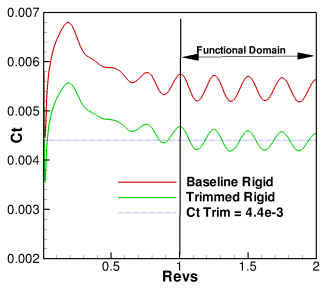
Figure 15: Residual convergence in a typical adjoint time step

root and tip twist definitions, and a bound of $\pm 5.0°$ of pitch angle is set on all the pitch parameters (collective and cyclics). The optimizations were performed on the Yellowstone supercomputer at the NCAR-Wyoming Supercomputing Center (NWSC) with the simulations (analysis/adjoint) running in parallel on 1024 cores. A typical coupled functional gradient (analysis/adjoint) computation step requires approximately 70 minutes when run on 1024 cores.

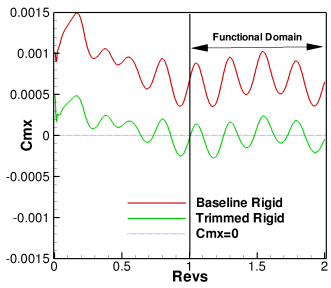
The performance optimization consists of three main stages: i) Trim ii) Shape/performance optimization and, iii) re-trim. The "Trim" step involves trimming the rotor to a target wind tunnel rotor thrust value of $C_T = 0.0044$ and zero average longitudinal and lateral moments ($C_{M_Y}, C_{M_X} = 0$). The objective function used in this step consists only of the thrust and moment terms in equation (150) ($\omega_1 = 0$) and the objective minimization is performed using only the three pitch parameters, namely, collective and cyclics as design variables.

In the second stage, blade shape optimization is performed by including the performance objective, i.e., C_Q term in the time-dependent objective function to be minimized. Appropriate weights ($\omega_i, i = 1, 2, 3, 4$) are used to maintain the rotor trim state through a penalty function while the blade shape is optimized to obtain minimum rotor power. In total 115 design parameters are used in this stage, including 112 blade shape parameters and 3 pitch control parameters. However, even after optimization convergence in stage two, the exact trim state is not maintained. This is because the trim objective components in the objective function are used only as weak constraint terms, i.e. as penalty terms. Therefore, the last stage involves re-trimming the rotor back to the target thrust and moment values, once again using only the three control pitch parameters as design inputs.

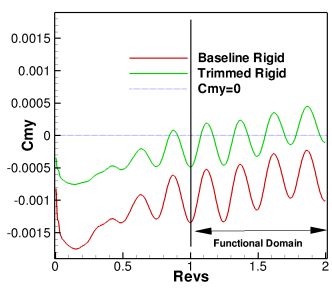
The optimization results for the rigid blade case are presented first, followed by the flexible blade optimization results. Figure 16 demonstrates that the rigid HART-II rotor is successfully trimmed to the target rotor thrust ($C_T = 0.0044$) and moment values ($C_{M_X}, C_{M_Y} = 0$). Figure 17.1 shows that the trim objective gradient drops by more than 5 orders of magnitude, while the objective functional achieves a minimum over 26 design iterations. Figure 17.2 further shows consistent convergence of all the three pitch parameters.



16.1: Thrust trim



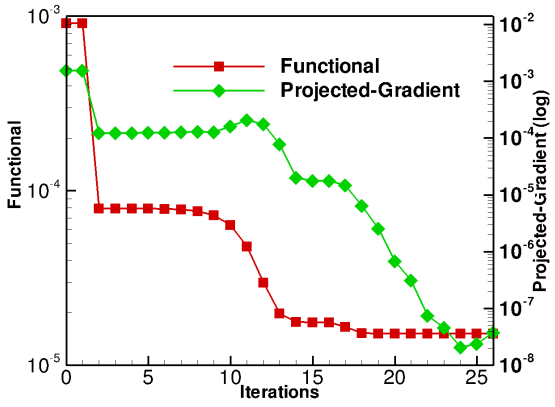
16.2: Lateral moment trim



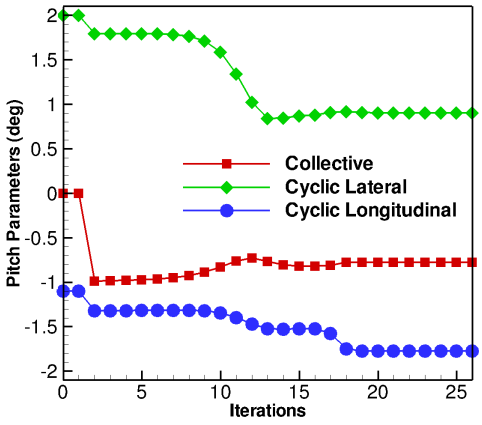
16.3: Longitudinal moment trim

Figure 16: HART-II rigid blade trim

Figures 18 summarizes the optimized rotor blade shape as well as re-trimmed rotor performance plots. Figure 18.1 shows that the baseline trimmed rigid rotor is shape



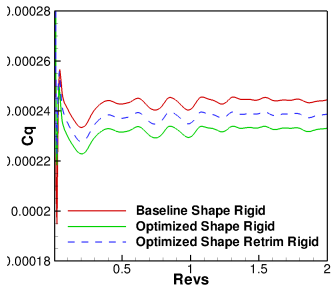
17.1: Trim optimization convergence



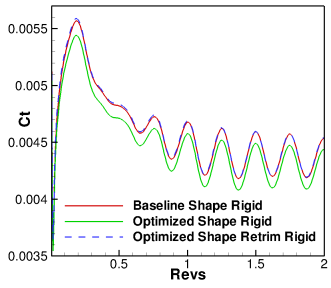
17.2: Pitch parameters convergence

Figure 17: HART-II rigid blade trim convergence

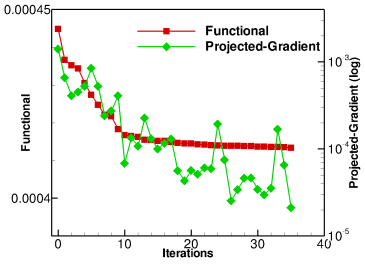
optimized to achieve approximately 4.7% power reduction with a penalty of approximately 2.3% thrust loss. Although some of this gain is lost upon retrim, the final optimized and trimmed rotor achieves an overall 2.8% reduction in power compared to the baseline blade. The near identical lines of the baseline and retrimmed rotor thrust plots in Figure 18.2 confirm that the rotor is retrimmed back to the trim target thrust value. Figure 18.3 shows 2 orders of magnitude gradient drop and consistent functional convergence after 35 design iterations for the blade shape optimization. Figure 19 shows the shape optimized blade sections at 9 stations. In general, the optimized blade shape results in thicker inboard and thinner outboard stations.



18.1: Power time history



18.2: Thrust time history



18.3: Optimization convergence

Figure 18: HART-II rigid blade shape optimization and re-trim

Figures 20 illustrates the trim optimization of the flexible HART-II rotor to the trim target thrust ($C_T = 0.0044$) and moment values ($C_{M_X}, C_{M_Y} = 0$). Figure 21.1 shows that the trim objective gradient drops by more than 2 orders of magnitude, while the objective functional achieves a minimum over 19 design iterations. Figure 21.2 further shows consistent convergence of all the three pitch parameters.

Figures 22 summarizes the optimized flexible blade shape as well as re-trimmed performance plots. Figure 22.1 shows the baseline trimmed flexible rotor is shape optimized to

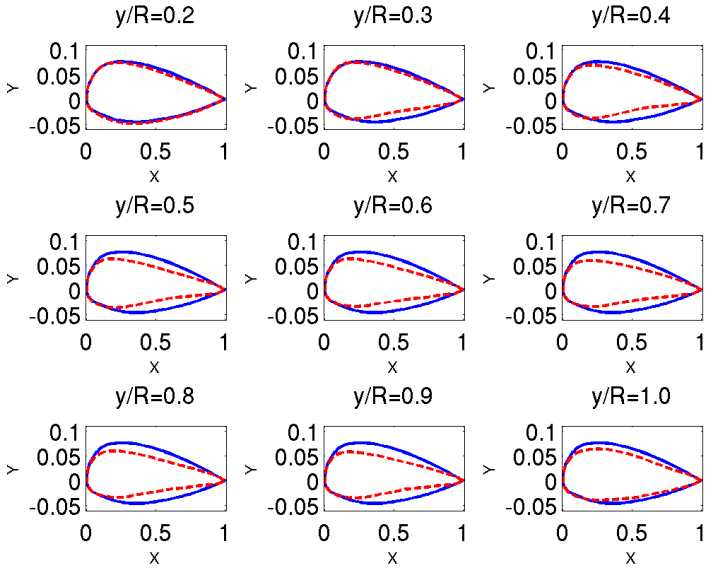
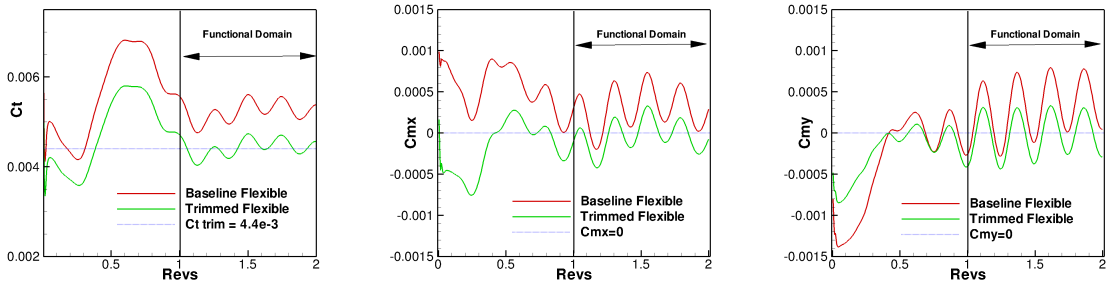
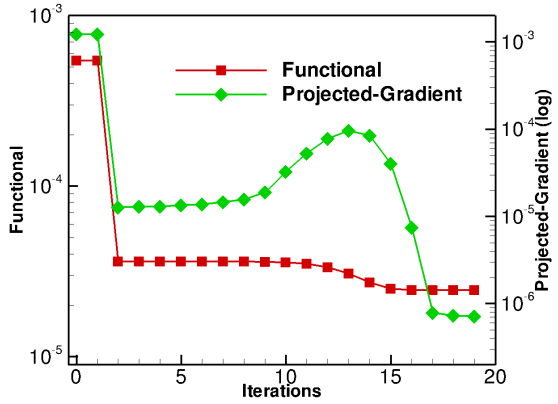


Figure 19: HART-II rigid blade optimized blade sections; solid-baseline, dashed-optimized

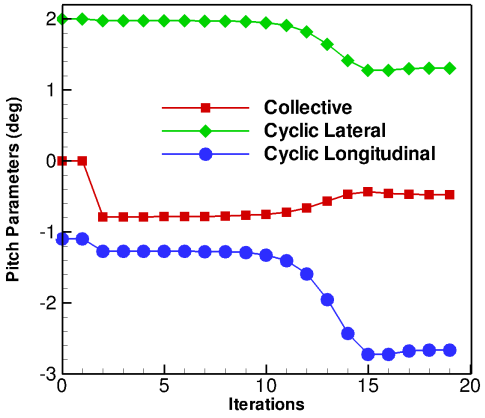


20.1: Thrust trim 20.2: Lateral moment trim 20.3: Longitudinal moment trim

Figure 20: HART-II flexible blade trim



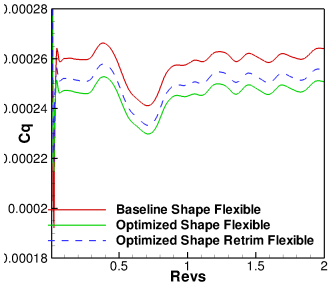
21.1: Trim optimization convergence



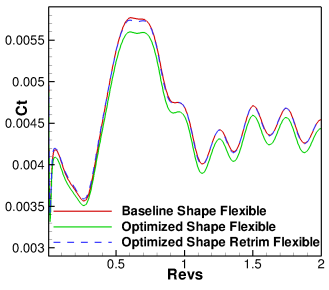
21.2: Pitch parameters convergence

Figure 21: HART-II flexible blade trim convergence

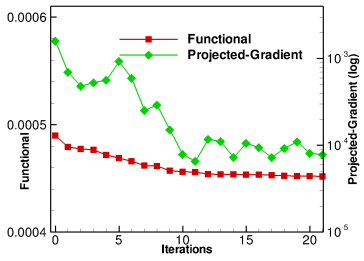
achieve a significant power reduction of approximately 5.0% with a thrust loss of approximately 2.6%. Similar to the rigid blade re-trim, for the flexible blade some of this gain is lost upon re-trim. However, the final shape optimized and re-trimmed rotor achieves an overall 3.1% reduction in power compared to the baseline blade. Figure 22.3 shows 1.5 orders of magnitude gradient drop and consistent functional convergence after 21 design iterations for the blade shape optimization stage. Figure 23 shows the shape optimized blade sections at 9 stations. The optimized blade shapes show similar trend to those observed in the rigid blade shape optimization, i.e. thicker inboard and thinner outboard stations.



22.1: Power time history



22.2: Thrust time history



22.3: Optimization convergence

Figure 22: HART-II flexible blade shape optimization and re-trim

6 Conclusions and Future Outlook

In this chapter, we have presented various formulations for deriving discrete forward and adjoint sensitivity methods for time-dependent and multidisciplinary problems. We have shown how time-dependent problems lead to a reverse recurrence relation in time in the

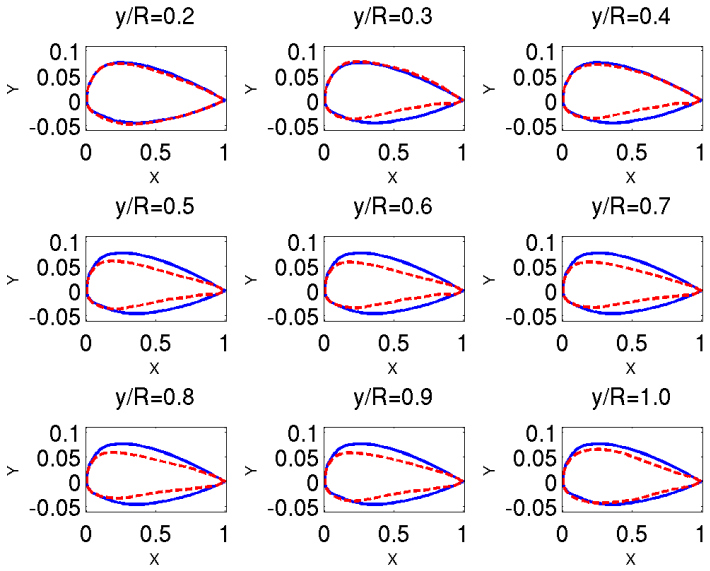


Figure 23: HART-II flexible blade optimized blade sections; solid-baseline, dashed-optimized

adjoint formulation for generic problems. We have also described in detail discrete tangent and adjoint implementations for finite-volume moving mesh computational fluid dynamics problems in arbitrary Lagrange Eulerian (ALE) form. Using a higher level compact notation, we have developed a framework for deriving the discrete tangent and adjoint formulations for time-dependent multidisciplinary problems in a manner that reuses disciplinary data-structures and solver strategies developed for the corresponding analysis problem. This approach was demonstrated on an aeroelastic rotor optimization problem using a beam structural model for the rotor blades.

Although the flexible rotor test case represents a realistic rotor optimization problem, there are various improvements that can be incorporated to improve the optimization outcomes. These include the use of structural as well as aerodynamic shape design parameters, the incorporation of additional constraints (both aerodynamic and structural), the use of more sophisticated shell and brick-based structural models, and the use of more sophisticated optimizers which handle constraints directly (as opposed to the penalty formulation used in this work). However, these techniques are related to the optimization process, which is simply one of the various possible applications for adjoint methods. One of the principal goals of this optimization example, and the associated gradient verification exercise, has been to show that the described tangent and adjoint formulations can be used for arbitrarily complex time-dependent multidisciplinary problems in an effective manner.

At the same time, several issues remain to be investigated in the context of time-dependent multidisciplinary tangent and adjoint methods. Firstly, although our verification exercise demonstrated machine precision accuracy for the coupled multidisciplinary adjoint and tangent sensitivities, the equations were fully converged at each time step in this case to avoid any contamination by algebraic error. In practice, it is not practical

to fully converge most time-dependent simulations at each time step. In recent work [23] we have examined the accuracy of these gradients in the presence of partial convergence and have noted that the errors in the computed gradients grow in time and can become significant for long time-integration problems. Thus it will be important in future work to better understand the effect algebraic error has on computed sensitivities and to develop convergence tolerance criteria for controlling such errors. This sensitivity error growth due to incomplete convergence is a separate issue from sensitivity divergence observed in chaotic systems [35] and is likely the dominant error source for the class of unsteady Reynolds-Averaged Navier-Stokes (RANS) simulations that pertain to problems such as time-dependent aeroelasticity. However, many time-dependent problems, especially those involving turbulence resolving simulations such as large-eddy simulations (LES) will likely display chaotic behavior and require the development of specialized techniques such as those described in [35] to obtain suitable sensitivities.

7 Acknowledgements

A significant portion of these notes is based on work performed by Dr. Karthik Mani and Dr. Asitav Mishra while they were in residence at the University of Wyoming, as well as contributions from Professor Jay Sitaraman. Computer resources were provided by the University of Wyoming Advanced Research Computing Center and by the NCAR-Wyoming Supercomputer Center.

References

[1] C. Bischof, A. Carle, P. Khademi, and A. Mauer. The ADIFOR 2.0 system for the automatic differentiation of Fortran 77 programs. Technical Report CRPC-TR94491, Center for Research on Parallel Computation, Rice University, 1994.

[2] J. A. Fike and J. J. Alonso. Automatic differentiation through the use of hyper-dual numbers for second derivatives. In *Recent Advances in Algorithmic Differentiation*, pages 163–173. Springer, 2012.

[3] P. Geuzaine, C. Grandmont, and C. Farhat. Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations. *Journal of Computational Physics*, 191:206–227, 2003.

[4] M. B. Giles and E. Suli. Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality. *Acta Numerica*, pages 145–236, 2002.

[5] R. Hartmann. Adjoint consistency analysis of discontinuous Galerkin discretizations. *SIAM Journal on Numerical Analysis*, 45(6):2671–2696, 2007.

[6] L. Hascoet. TAPENADE: A tool for automatic differentiation of programs. In *Proceedings of the 4th European Conference on Computational Mechanics, ECCOOMAS' 2004, Jyvaskyla, Finland*, 2004.

[7] A. Jameson. Aerodynamic shape optimization using the adjoint method. VKI Lecture Series on Aerodynamic Drag Prediction and Reduction, von Karman Institute of Fluid Dynamics, Rhode St Genese, Belgium, February 2003.

[8] A. Jameson, J. J. Alonso, J. J. Reuther, L. Martinelli, and J. C. Vassberg. Aerodynamic shape optimization techniques based on control theory. AIAA Paper 98-2538, June 1998.

[9] G. Kenway and J. R. R. A. Martins. Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration. *Journal of Aircraft*, 51(1):144–160, 2014.

[10] J. N. Lyness. Numerical algorithms based on the theory of complex variables. Proceedings ACM 22nd National Conference, Thomas Book Company, Washington DC, 1967.

[11] K. Mani, A. Mishra, D. J. Mavriplis, and J. Sitaraman. Helicopter rotor design using adjoint-based optimization in a coupled CFD-CSD framework. AHS 69 Forum paper, Phoenix AZ, May 2013.

[12] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. A coupled-adjoint sensitivity analysis method for high-fidelity aero-structural design. *Optimization and Engineering*, 6(1):33–62, 2005.

[13] K Maute, M Nikbay, and C Farhat. Coupled analytical sensitivity analysis and optimization of three-dimensional nonlinear aeroelastic systems. *AIAA journal*, 39(11):2051–2061, 2001.

[14] D. J. Mavriplis. Directional agglomeration multigrid techniques for high-Reynolds number viscous flow solvers. *AIAA Journal*, 37(10):1222–1230, October 1999.

[15] D. J. Mavriplis. A discrete adjoint-based approach for optimization problems on three dimensional unstructured meshes. *AIAA Journal*, 45(4):741–750, April 2007.

[16] D. J. Mavriplis. Solution of the unsteady discrete adjoint for three- dimensional problems on dynamically deforming unstructured meshes. AIAA Paper 2008-0727, 46th Aerospace Sciences Meeting, Reno NV, January 2008.

[17] D. J. Mavriplis. Third drag prediction workshop results using the NSU3D unstructured mesh solver. *AIAA Journal of Aircraft*, 45(3):750–761, May 2008.

[18] D. J. Mavriplis, M. Long, T. Lake, and M. Langlois. NSU3D results for the second AIAA High-lift Prediction Workshop. *Journal of Aircraft*, pages 1–19, 2015.

[19] D. J. Mavriplis and K. Mani. Unstructured mesh solution techniques using the NSU3D solver. AIAA Paper 2014-081, 52nd Aerospace Sciences Meeting, National Harbor, MD, January 2014.

[20] D. J. Mavriplis and S. Pirzadeh. Large-scale parallel unstructured mesh computations for 3D high-lift analysis. *AIAA Journal of Aircraft*, 36(6):987–998, December 1999.

[21] D. J. Mavriplis and Z. Yang. Construction of the discrete geometric conservation law for high-order time accurate simulations on dynamic meshes. *Journal of Computational Physics*, 213(2):557–573, April 2006.

[22] A. Mishra, K. Mani, D. J. Mavriplis, and J. Sitaraman. Helicopter rotor design using adjoint-based optimization in a coupled cfd-csd framework. In *69th American Helicopter Society Annual Forum, Phoenix, AZ*, May 21–23 2013.

[23] A. Mishra, K. Mani, D. J. Mavriplis, and J. Sitaraman. Multipoint time-dependent aeroelastic adjoint-based aerodynamic shape optimization of helicopter rotors. In *71st American Helicopter Society Annual Forum, Virginia Beach, VA*, May 5–7 2015.

[24] A. Mishra, D. J. Mavriplis, and J. Sitaraman. Time-dependent aero-elastic adjoint-based aerodynamic shape optimization of helicopter rotors in forward flight. AIAA Paper 2015-1130, 56th AIAA Aerospace Sciences Meeting, Kissimmee, FL, January 2015.

[25] E. J. Nielsen, J. Lu, M. A. Park, and D. L. Darmofal. An implicit exact dual adjoint solution method for turbulent flows on unstructured grids. *Computers and Fluids*, 33(9):1131–1155, November 2004.

[26] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Optimization Research, 2000.

[27] T. A. Oliver and D. Darmofal. Analysis of dual consistency for discontinuous Galerkin discretizations of source terms. *SIAM Journal on Numerical Analysis*, 47(5):3507–3525, 2009.

[28] J. J. Reuther, J. J. Alonso, J. R. R. A. Martins, and S. C. Smith. A coupled aero-structural optimization method for complete aircraft configurations. AIAA Paper 97-187, 37th Aerospace Sciences Meeting, Reno, NV, January 1999.

[29] D. M. Schuster, P. Chwalowski, J. Heeg, and C. D. Wieseman. Summary of data and findings from the first aeroelastic prediction workshop. Paper ICCFD7-3101, Proceedings of the 7th ICCFD Conference, Mauna Lani Bay Hawaii, July 2012.

[30] A. Sei and W. W. Symes. A note on consistency and adjointness for numerical schemes. Technical Report TR95-04, Department of Computational and Applied Mathematics, Rice University, 1995.

[31] J. Sitaraman, M. Potsdam, A. Wissink, B. Jayaraman, D. Mavriplis A. Datta, and H. Saberi. Rotor loads prediction using HELIOS: A multisolver framework for rotorcraft aeromechanics analysis. *Journal of Aircraft*, 50(2):478–492, 2013.

[32] M. J. Smith, J. W. Lim, B. G. van der Wall, J. D. Baeder, R. T. Bierdron, D. D. Boyd Jr., B. Jayaraman, S. N. Junk, and B-Y Min. Adjoint-based unsteady airfoil design optimization with application to dynamic stall. In *68th American Helicopter Society Annual Forum, Fort Worth, TX*, May 1–3 2012.

[33] P. R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. *La Recherche Aérospatiale*, 1:5–21, 1994.

[34] P. D. Thomas and C. K. Lombard. Geometric conservation law and its applications to flow computations on moving grids. *AIAA J.*, 17:1030–1037, 1979.

[35] Q. Wang. Forward and adjoint sensitivity computation of chaotic dynamical systems. *Journal of Computational Physics*, 235:1–13, 2013.

[36] Z. Yang and D. J. Mavriplis. A mesh deformation strategy optimized by the adjoint method on unstructured meshes. *AIAA Journal*, 45(12):2885–2896, 2007.

[37] Y. H. Yu, C. Tung, B. van der Wall, H-J. Pausder, C. Burley, T. Brooks, P. Beaumier, Y. Delrieux, E. Mercker, and K. Pengel. The HART-II test: Rotor wakes and aeroacoustics with higher-harmonic pitch control (hhc) inputs -the joint german/french/dutch/us project-. In *58th American Helicopter Society Annual Forum, Montreal, Canada*, June 11–13 2002.

[38] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. LBFGS-B: A limited memory FORTRAN code for solving bound constrained optimization problems. Technical Report NAM-11, EECS Department, Northwestern University, 1994.